RESTful Network API for Chat

Candidate Version 1.0 – 09 May 2012

Open Mobile Alliance

OMA-TS-REST_NetAPI_Chat-V1_0-20120509-C

Contents

Figures

Tables

1. Scope

This specification defines a RESTful API for Chat using HTTP protocol bindings.

2. References

2.1 Normative References

[Autho4API_10] "Authorization Framework for Network APIs", Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL: http://www.openmobilealliance.org/

[IETF_ACR_draft] "The acr URI for anonymous users", S.Jakobsson, K.Smith, July 2011, URL: http://tools.ietf.org/html/draft-uri-acr-extension-03

[RC_API_RD] APIs for Rich Communications Requirements, OMA-RD-RC_API-V1_0, Open Mobile Alliance, URL: http://www.openmobilealliance.org/

[REST_NetAPI_Common] "Common definitions for RESTful Network APIs", Open Mobile Alliance™, OMA-TS-REST_NetAPI_Common-V1_0, URL: http://www.openmobilealliance.org/

[REST_NetAPI_NotificationChannel] "RESTful Network API for Notification Channel", Open Mobile Alliance™, OMA-TS-REST_NetAPI_NotificationChannel-V1_0, URL: http://www.openmobilealliance.org/

[REST_SUP_Chat] "XML schema for the RESTful Network API for Chat", Open Mobile Alliance™, OMA-SUP-XSD-rest_netapi_chat-V1_0, URL: http://www.openmobilealliance.org/

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL: http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] "Hypertext Transfer Protocol -- HTTP/1.1", R. Fielding et. al, January 1999, URL: http://www.ietf.org/rfc/rfc2616.txt

[RFC3261] "SIP: Session Initiation Protocol", J. Rosenberg et al., June 2002, URL: http://www.ietf.org/rfc/rfc3261.txt

[RFC3966] "The tel URI for Telephone Numbers", H. Schulzrinne, December 2004, URL: http://www.ietf.org/rfc/rfc3966.txt

[RFC3986] "Uniform Resource Identifier (URI): Generic Syntax", R. Fielding et. al, January 2005, URL: http://www.ietf.org/rfc/rfc3986.txt

[RFC4627] "The application/json Media Type for JavaScript Object Notation (JSON)", D. Crockford, July 2006, URL: http://www.ietf.org/rfc/rfc4627.txt

[RFC4975] "The Message Session Relay Protocol (MSRP)", B. Campbell et. al, September 2007, URL: http://www.ietf.org/rfc/rfc4975.txt

[RFC5438] "Instant Message Disposition Notification (IMDN)", E. Burger and H. Khartabil, September 2007, URL: http://www.ietf.org/rfc/rfc5438.txt

[SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[W3C_URLENC] HTML 4.01 Specification, Section 17.13.4 Form content types, The World Wide Web Consortium, URL: http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.1

[XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL: http://www.w3.org/TR/xmlschema-1/

[XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes Second Edition, URL: http://www.w3.org/TR/xmlschema-2/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/

[REST_WP]	"Guidelines for RESTful Network APIs", Open Mobile Alliance™, OMA-WP-Guidelines_for_RESTful_Network_APIs, URL: http://www.openmobilealliance.org/
[RFC3994]	„Indication of Message Composition for Instant Messaging", H. Schulzrinne, January 2005, URL: http://www.ietf.org/rfc/rfc3994.txt
[RFC4575]	"A Session Initiation Protocol (SIP) Event Package for Conference State", J. Rosenberg et. al, August 2006, URL: http://www.ietf.org/rfc/rfc3994.txt
[SIMPLE_IM]	"Instant Messaging using SIMPLE ", Open Mobile Alliance™, OMA-TS-SIMPLE_IM-V1_0, URL: http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Ad-hoc 1-1 Chat	A one-to-one chat that allows exchanging messages and reporting message status without the need and ability to manage chat sessions.
Client-side Notification URL	An HTTP URL exposed by a client, on which it is capable of receiving notifications and that can be used by the client when subscribing to notifications.
Confirmed 1-1 Chat	A one-to-one chat that requires opening and confirming a session before it allows exchanging messages and reporting message status. Confirmed 1-1 Chats also allow terminating the session.
Long Polling	A variation of the traditional polling technique, where the server does not reply to a request unless a particular event, status or timeout has occurred. Once the server has sent a response, it closes the connection, and typically the client immediately sends a new request. This allows the emulation of an information push from a server to a client.
Notification Channel	A channel created on the request of the client and used to deliver notifications from a server to a client. The channel is represented as a resource and provides means for the server to post notifications and for the client to receive them via specified delivery mechanisms. For example in the case of Long Polling the channel resource is defined by a pair of URLs. One of the URLs is used by the client as a call-back URL when subscribing for notifications. The other URL is used by the client to retrieve notifications from the Notification Server.
Notification Server	A server that is capable of creating and maintaining Notification Channels.
Originator	The party that initiates a chat session.
Participant	A party that participates in a chat session, including the Originator.
Receiver	The party that receives a chat message.
Sender	The party that sends a chat message.
Server-side Notification URL	An HTTP URL exposed by a Notification Server, that identifies a Notification Channel and that can be used by a client when subscribing to notifications.
Terminating Participant	A Participant in a chat session that is not the Originator.

Additionally, all definitions from the OMA Dictionary apply [OMADICT].

3.3 Abbreviations

ACR	Anonymous Customer Reference
API	Application Programming Interface
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
OMA	Open Mobile Alliance

REST	REpresentational State Transfer
SCR	Static Conformance Requirements
SIP	Session Initiation Protocol
TS	Technical Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification of the RESTful Network API for Chat contains HTTP protocol bindings based on the requirements for Chat (also known as Instant Messaging) defined in [RC_API_RD], using the REST architectural style. The specification provides resource definitions, the HTTP verbs applicable for each of these resources, and the element data structures, as well as support material including flow diagrams and examples using the various supported message body formats (i.e. XML, JSON, and application/x-www-form-urlencoded).

4.1 Version 1.0

Version 1.0 of this specification supports the following operations:

- Managing subscriptions to chat-related event notifications
- Sending and receiving 1-1 chat messages
- Reporting the status of 1-1 chat messages
- Receiving notifications about the status of 1-1 chat messages
- Managing 1-1 chat sessions
- Promoting a 1-1 chat session into a group chat session
- Managing group chat sessions
- Sending and receiving group chat messages
- Receiving notifications about invitations to a 1-1 chat
- Receiving notifications about invitations to a group chat
- Receiving notifications about chat session events
- Receiving notifications about a change in the list of Participants in a group chat session

In addition, this specification provides:

- Support for scope values used with authorization framework defined in [Autho4API_10]
- Support for Anonymous Customer Reference (ACR) as an end user identifier
- Support for “acr:Authorization” as a reserved keyword in a resource URL variable that identifies an end user

5. Chat API definition

This section is organized to support a comprehensive understanding of the Chat API design. It specifies the definition of all resources, definition of all data structures, and definitions of all operations permitted on the specified resources.

Common data types, naming conventions, fault definitions and namespaces are defined in [REST_NetAPI_Common].

The remainder of this document is structured as follows:

Section 5 starts with a diagram representing the resources hierarchy, followed by a table listing all the resources (and their URL) used by this API, along with the data structure and the supported HTTP verbs (section 5.1). What follows are the data structures (section 5.2). A sample of typical use cases is included in section 5.2.3, described as high level flow diagrams.

Section 6 contains the detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP commands, the possible HTTP response codes, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 6 use XML as the format for the message body. Application/x-www-form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D. Appendix B provides the Static Conformance Requirements (SCR).

Appendix E provides the operations mapping to a pre-existing baseline specification, where applicable. Appendix F provides a list of all lightweight resources, where applicable.

Appendix G defines authorization aspects to control access to the resources defined in this specification.

Note: Throughout this document client and application can be used interchangeably.

5.1 Resources Summary

This section summarizes all the resources used by the RESTful Network API for Chat.

The "apiVersion" URL variable SHALL have the value "v1" to indicate that the API corresponds to this version of the specification. See [REST_NetAPI_Common] which specifies the semantics of this variable.

The figure below visualizes the resource structure defined by this specification. Note that those nodes in the resource tree which have associated HTTP methods defined in this specification are depicted by solid boxes.

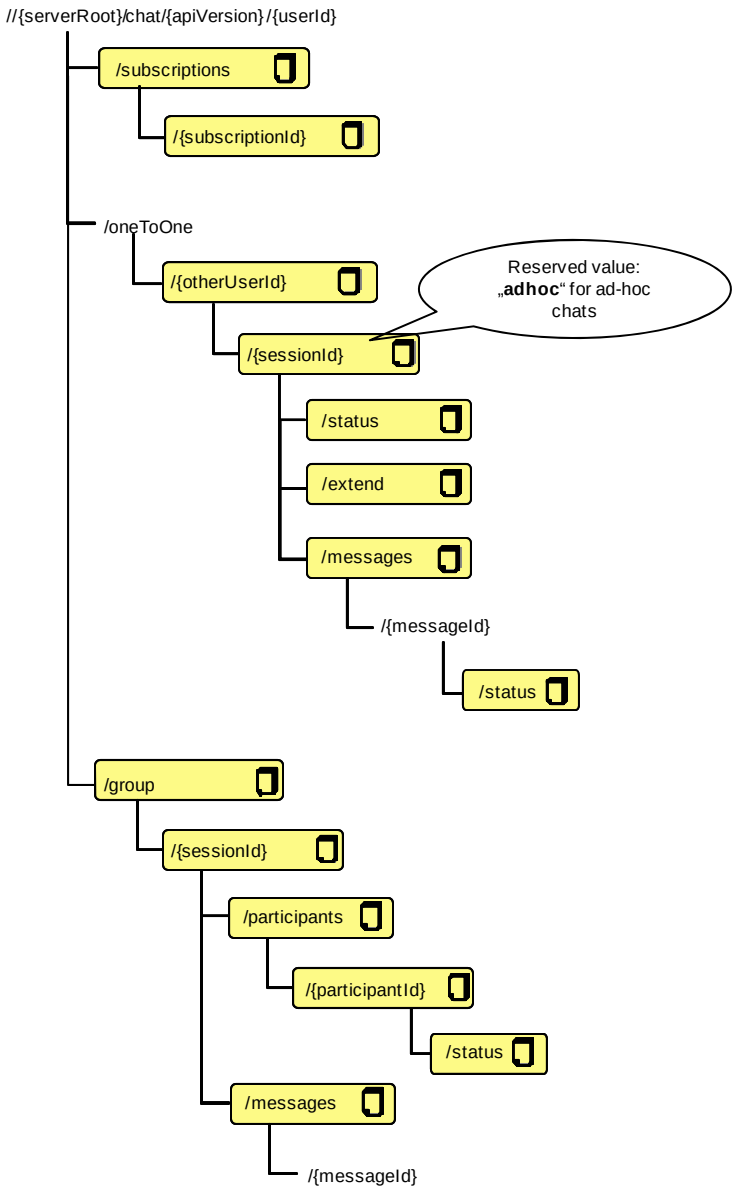

Figure 1 Resource structure defined by this specification

The following tables give a detailed overview of the resources defined in this specification, the data type of their representation and the allowed HTTP methods.

Note: 1-1 chats (Ad-hoc or Confirmed) and group chats represent different approaches and are therefore treated differently.

In an Ad-hoc 1-1 Chat, the Participants interact direct and spontaneous without the need to create a session at API level. For that purpose, they access the resource with {sessionId} set to the reserved value “adhoc”. The underlying network layers still need to set up a session. In an Ad-hoc 1-1 Chat, the Originator can send messages to the Terminating Participant during the set-up phase of the session, as well as when the session has been established. Implementations of Ad-hoc 1-1 Chats MAY support the GET method on the {sessionId} node with the value “adhoc”. No session-management functionality is exposed via the API.

In a Confirmed 1-1 Chat, the two phases of session set-up and in-session communication are clearly separated. First, the Originator requests the creation of a chat session, which starts the session set-up phase in the underlying network layers by sending an offer to the Terminating Participant to enter a chat. If the Terminating Participant confirms that offer, the session is established. During the session set-up phase, no messages can be exchanged at the underlying network layers; message exchange only takes place in-session. Implementations MAY buffer messages sent during session set-up phase and send them once the session has been established, or MAY reject such an attempt with HTTP response code 403 and a POL1012 exception (see section 7). Implementations of Confirmed 1-1 Chats MUST support the GET and DELETE methods on the {sessionId} node.

In a group chat session, a chat server (called “conference focus” in [SIMPLE_ IM]) is involved in the communication that filters and aggregates the traffic, and each Participant is connected to the conference focus using a session model. This architecture results in different handling of many of the events, and also in different sets of events available. In order to provide a clean separation of these different feature sets, 1-1 chat and group chat are modeled as different sets of resources. A 1-1 chat can incorporate exactly 2 Participants, whereas a group chat can incorporate one Originator and one or more Terminating Participants. The different types of 1-1 chat (Ad-hoc vs. Confirmed) allow to be mapped to different underlying systems that exist in the market.

Purpose: Allow the client to manage subscriptions for chat notifications

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to chat event notifications	/subscriptions	ChatSubscriptionList (used for GET) ChatNotificationSubscription (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	Read the list of active chat notification subscriptions	no	Create new subscription to chat notifications	no
Individual subscription to chat event notifications	/subscriptions/{subscriptionId}	ChatNotificationSubscription	Read an individual chat notification subscription	no	no	Cancel subscription and stop corresponding notifications

Purpose: Allow the client to handle 1-1 chats

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 chat sessions between two users	/{otherUserId}	ChatSessionInformation (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Create a 1-1 chat session	no

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Individual 1-1 chat session	/{otherUserId}/{sessionId}	ChatSessionInformation	Read 1-1 chat session information	no	no	Cancel invitation (Originator) Decline invitation (Terminating Participant) Terminate session
1-1 chat session status	/{otherUserId}/{sessionId}/status	ParticipantSessionStatus	no	Accept a 1-1 chat session invitation	no	no
Extend 1-1 chat to a group chat session	/{otherUserId}/{sessionId}/extend	ParticipantList	no	no	Extend a 1-1 chat session to a group chat session	no

Purpose: Allow the client to handle 1-1 chat messages

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Chat messages in a 1-1 chat	/messages	ChatMessage (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Create (send) a chat message	no
Individual message status in a 1-1 chat	/messages/{messageId}/status	MessageStatusReport	Read the status of a chat message	Report the status of a chat message	no	no

Purpose: Allow the client to handle group chat sessions

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All group chat sessions	/group	GroupChatSessionInformation (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Create a new group chat session	no
Individual group chat session	/group/{sessionId}	GroupChatSessionInformation	Retrieve group chat session information	no	no	Terminate group chat session (Originator)

Purpose: Allow the client to handle group chat Participants

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All Participants in a group chat session	/participants	ParticipantList (used for GET) ParticipantList or ParticipantInformation (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	Read the list of group chat Participants	no	Add one or more group chat Participant(s) Re-join session	No
Individual Participant in a group chat session	/participants/{participantId}	ParticipantInformation	Read information about an individual group chat Participant	no	no	Remove Participant from group chat session (Originator) Decline invitation (Terminating Participant) Leave session (Participant)
Individual group chat session Participant status	/participants/{participantId}/status	ParticipantSessionStatus		Accept group chat session invitation	no	no

Purpose: Allow the client to handle group chat messages

Resource	URL	Data Structures	HTTP verbs

	Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}		GET	PUT	POST	DELETE
Chat messages in a group chat session	/messages	ChatMessage (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Create a chat message	no

Purpose: Allow the client to receive chat notifications

Resource	URL Base URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification containing incoming message	Specified by client when subscription is created or provisioned	MessageNotification	no	no	Notify client about incoming chat message	no
Client notification about message status	Specified by client when subscription is created or provisioned	MessageStatusNotification	no	no	Notify client about the status of a chat message it has sent	no
Client notification about 1-1 chat session invitations	Specified by client when subscription is created or provisioned	SessionInvitationNotification	no	no	Notify client about incoming 1-1 chat invitation	no

Resource	URL Base URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification about group chat session invitations	Specified by client when subscription is created or provisioned	GroupSessionInvitationNotification	no	no	Notify client about incoming group chat invitation	no
Client notification about chat session events	Specified by client when subscription is created or provisioned	ChatEventNotification	no	no	Notify client about chat events	no
Client notification about changes of Participant status	Specified by client when subscription is created or provisioned	ParticipantStatusNotification	no	no	Notify client about Participant status changes	no
Client notification about subscription cancellation	Specified by client when subscription is created or provisioned	SubscriptionCancellationNotification	no	no	Notify client that a subscription has been cancelled (e.g. expired)	no

5.2 Data Types

5.2.1 XML Namespaces

The XML namespace for the Chat API data types is:

urn:oma:xml:rest:netapi:chat:1

The 'xsd' namespace prefix is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace prefix is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of namespace prefixes such as 'xsd' is not semantically significant.

The XML schema for the data structures defined in the section below is given in [REST_SUP_Chat].

5.2.2 Structures

The subsections of this section define the data structures used in the Chat API.

Some of the structures can be instantiated as so-called root elements.

For structures that contain elements which describe a user identifier, the statements in section 6 regarding 'tel', 'sip' and 'acr' URI schemes apply.

5.2.2.1 Type: ChatSubscriptionList

This type represents a list of chat notification subscriptions.

Element	Type	Optional	Description
chatNotificationSubscription	ChatNotificationSubscription [0..unbounded]	Yes	Array of chat notification subscriptions
resourceURL	xsd:anyURI	No	Self referring URL

A root element named chatSubscriptionList of type ChatSubscriptionList is allowed in response bodies.

5.2.2.2 Type: ChatNotificationSubscription

This type represents a subscription to chat-related event notifications, i.e. all notifications of type ChatEventNotification, SessionInvitationNotification, ParticipantStatusNotification, and MessageNotification, GroupSessionInvitationNotification and MessageStatusNotification targeted at a particular user.

Element	Type	Optional	Description
callbackReference	common:CallbackReference	No	Client's Notification URL and OPTIONAL callbackData
confirmedChatSupported	xsd:boolean	Yes	In resource-creating requests, this flag signals to the server that this client supports Confirmed 1-1 Chats. In case this is present and set to

			true, the client supports Confirmed 1-1 Chats. In the created resource, the server sets this flag to true in case it was set to true by the client in the corresponding creation request and the server supports Confirmed 1-1 Chats; otherwise the server either sets it to false or omits it. If the server does not support any of the methods signalled by the client using the elements “confirmedChatSupported” and “adhocChatSupported”, it SHALL reject the subscription either with the exception POL1013 (if Confirmed 1-1 Chats are not supported) or with the exception POL1014 (if session-Ad-hoc 1-1 Chats are not supported). Default: false
adhocChatSupported	xsd:boolean	Yes	In resource-creating requests, this flag signals to the server that this client supports Ad-hoc 1-1 Chats. In case this is absent or set to true, the client supports for Ad-hoc 1-1 Chats. In the created resource, the server sets this flag to true or omits it in case it was absent or set to true in the corresponding creation request and the server supports Ad-hoc 1-1 Chats; otherwise the server sets it to false. If the server does not support any of the methods signalled by the client using the elements “confirmedChatSupported” and “adhocChatSupported”, it SHALL reject the subscription either with the exception POL1013 (if Confirmed 1-1 Chats are not supported) or with the exception POL1014 (if Ad-hoc 1-1 Chats are not supported). Default: true Note: the default is “true” here for maximum simplification of the API.
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to “0” (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the desired lifetime of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.

clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscriptions in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named chatNotificationSubscription of type ChatNotificationSubscription is allowed in request and/or response bodies.

Note that the clientCorrelator is used for purposes of error recovery as specified in [REST_NetAPI_Common], and internal client purposes. The server is NOT REQUIRED to use the clientCorrelator value in any form in the creation of the URL of the resource. The document [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

5.2.2.3 Type: ChatEventNotification

This type represents a notification about chat events that only need to convey the type of event without additional type-specific parameters.

More specific notification types are defined below.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The ‘callbackData’ element if it was passed by the application in the ‘callbackReference’ element when creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) Depending on the value of eventType, the server MUST include links as defined by the

			actual Notification resource in section 6.19. Further, the server SHOULD include a link to the related subscription.
eventType	EventType	No	Type of event
eventDescription	xsd:string	Yes	Textual description of the event

A root element named chatEventNotification of type ChatEventNotification is allowed in notification request bodies.

5.2.2.4 Type: SessionInvitationNotification

This type represents the notification for a 1-1 chat session invitation.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server MUST include links as defined by the actual Notification resource in section 6.17. Further, the server SHOULD include a link to the related subscription.
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
originatorAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator
originatorName	xsd:string	Yes	Human readable name of the Originator
tParticipantAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Terminating Participant
tParticipantName	xsd:string	Yes	Human readable name of the Terminating Participant

A root element named sessionInvitationNotification of type SessionInvitationNotification is allowed in notification request bodies.

The recipient can accept the request by updating the status, which is addressed by the URL passed in the "href" attribute of the "link" element with rel="ParticipantSessionStatus".

Typically this URL is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/status

The recipient can decline the request by sending a DELETE request to the URL passed in the "href" attribute of the "link" element with rel="ChatSessionInformation".

Typically this URL is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}

This type is not relevant in group chats.

If the recipient fails to react within a time interval defined by service policies, the session invitation will time out. In case of a 1-1 session, this means that the session will terminate.

5.2.2.5 Type: GroupSessionInvitationNotification

This type represents a notification for a group chat session invitation.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server MUST include links as defined by the actual Notification resource in section 6.18. Further, the server SHOULD include a link to the related subscription.
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
participant	ParticipantInformation [2..unbounded]	No	Contains the list of Participants of the session.

A root element named groupSessionInvitationNotification of type GroupSessionInvitationNotification is allowed in notification request bodies.

Each recipient can accept the request by updating the status, which is addressed by the URL passed in the "href" attribute of the "link" element with rel="ParticipantSessionStatus".

Typically this URL is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}/status

The recipient can decline the request by sending a DELETE request to the URL passed in the "href" attribute of the "link" element with rel="ParticipantInformation".

Typically this URL is

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}

If the recipient fails to react within a time interval defined by service policies, the session invitation will time out. In case of a group session, this means that this recipient will not be mentioned in any ParticipantStatusNotification.

This type is not relevant in 1-1 chats.

5.2.2.6 Type: MessageNotification

This type represents a notification delivering an incoming chat message.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server MUST include links as defined by the actual Notification resource in section 6.15. Further, the server MAY include a link to the related subscription.
senderAdress	xsd:anyURI	No	Identifier of the Participant that sent the message (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
senderName	xsd:string	Yes	Name of the Sender
chatMessage	ChatMessage	Choice	The actual message
isComposing	IsComposing	Choice	"isComposing" message
dateTime	xsd:dateTime	Yes	The time when the message was sent

XSD modelling uses a "choice" to select either chatMessage or isComposing.

A root element named messageNotification of type MessageNotification is allowed in notification request bodies.

In case the "chatMessage" element contains the element "reportRequest", the recipient MUST acknowledge the requested event 'Displayed' by sending a PUT request with a "MessageStatusReport" root element in the body to the URL passed in the "href" attribute of the "link" element with rel="MessageStatusReport".

For 1-1 chat this URL is typically:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/messages/{messageId}/status.

For group chat this functionality is not supported.

5.2.2.7 Type: ParticipantStatusNotification

This type represents the Participant status notification.

It is used to inform about Participant status changes in a group chat.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when

			creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server MUST include links as defined by the actual Notification resource in section 6.20. Further, the server SHOULD include a link to the related subscription.
participant	ParticipantStatusEntry [1..unbounded]	No	The list of Participants At least those that changed status since the last notification MUST be included.

A root element named participantStatusNotification of type ParticipantStatusNotification is allowed in notification request bodies.

Note: This type is not relevant in 1-1 chats.

5.2.2.8 Type: ParticipantStatusEntry

This type represents the status of a chat Participant.

Element	Type	Optional	Description
address	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Participant
name	xsd:string	Yes	Human readable name of the Participant
status	ParticipantStatus	Yes	Connection status of the Participant
yourown	xsd:boolean	Yes	If present and set to true, this indicates that the status entry represents the Participant to which this data structure is sent in a message.
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server SHOULD include a link to the resource representing the Participant in the chat session.

5.2.2.9 Type: MessageStatusNotification

This type represents a notification about the status of a chat message.

Element	Type	Optional	Description

callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to notifications about chat events See [REST_NetAPI_Common]
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related chat session) The server MUST include links as defined by the actual Notification resource in section 6.16. Further, the server MAY include a link to the related subscription.
status	MessageStatus	No	Indicates the status of the message
errorCode	xsd:string	Yes	Code of the error, if any
description	xsd:string	Yes	Description of the error, if any

A root element named messageStatusNotification of type MessageStatusNotification is allowed in notification request bodies.

Note: This type is not relevant in group chats.

5.2.2.10 Type: ChatMessage

This type represents a chat message.

Element	Type	Optional	Description
text	xsd:string	No	Text content of a chat message
reportRequest	MessageStatus [0..unbounded]	Yes	List of status events to report This element is not relevant in group chats.
resourceURL	xsd:anyURI	Yes	Self referring URL The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests. Note that in this version of the specification, the resourceURL is only used for correlation purposes, as there is no HTTP method defined for this URL.

A root element named chatMessage of type ChatMessage is allowed in request bodies.

5.2.2.11 Type: MessageStatusReport

This type represents a response to a chat message notification.

It is only needed if the chat message includes an indication that the Sender wishes to receive a report about "Displayed" message status.

Note that the report regarding the "Delivered" message status is generated in the API Server by procedures of the underlying protocol layers which are out of scope of this specification.

Element	Type	Optional	Description
status	MessageStatus	No	Indicates the status of the message

A root element named messageStatusReport of type MessageStatusReport is allowed in request/response bodies.

Note: This type is not relevant in group chats.

5.2.2.12 Type: ParticipantSessionStatus

This type represents the status of a Participant in the chat session.

Element	Type	Optional	Description
status	ParticipantStatus	No	Status of the Participant To indicate that the client accepts the session invitation, this element MUST be set to "Connected". The client is not allowed to use in requests the remaining values of the ParticipantStatus enumeration. If the client uses one of these in a request, the server SHOULD respond with an HTTP status code "400 Bad request" and return a SVC0003 exception with the list of valid values set to "Connected".

A root element named participantSessionStatus of type ParticipantSessionStatus is allowed in request and/or response bodies.

5.2.2.13 Type: ChatSessionInformation

This type represents information about a 1-1 chat session.

Element	Type	Optional	Description
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
originatorAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator. If originatorAddress is also part of the request URL, the two MUST have the same value.
originatorName	xsd:string	Yes	Human readable name of the Originator

tParticipantAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Terminating Participant If tParticipantAddress is also part of the request URL, the two MUST have the same value.
tParticipantName	xsd:string	Yes	Human readable name of the Terminating Participant
status	ParticipantStatus	Yes	Connection status of the Participant Set by the server SHALL NOT be present in request bodies during resource creation
resourceURL	xsd:anyURI	Yes	Self referring URL The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named chatSessionInformation of type ChatSessionInformation is allowed in request and/or response bodies.

This type is not relevant in group chats.

5.2.2.14 Type: GroupChatSessionInformation

This type represents information about a group chat session.

Element	Type	Optional	Description
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
participant	ParticipantInformation [1..unbounded]	No	The Participant(s) connected or invited to this chat session
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate chat session creations in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL

			The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named groupChatSessionInformation of type groupChatSessionInformation is allowed in request and/or response bodies.

Regarding the clientCorrelator field, the note in section 5.2.2.2 applies.

This type is not relevant in 1-1 chats.

5.2.2.15 Type: ParticipantList

This type represents a list of chat Participants.

Element	Type	Optional	Description
participant	ParticipantInformation [1..unbounded]	No	List of chat Participants.
resourceURL	xsd:anyURI	Yes	Self referring URL The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named participantList of type ParticipantList is allowed in request and/or response bodies.

This type is not relevant in 1-1 chats.

5.2.2.16 Type: ParticipantInformation

This type represents a chat Participant.

It is based on the [RFC4575] as defined in [SIMPLE_IM] chapter 7.2.1.12.

Element	Type	Optional	Description
address	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Participant. If address is also part of the request URL, the two

			MUST have the same value.
name	xsd:string	Yes	Human readable name
isOriginator	xsd:boolean	Yes	If the Participant represented by this data structure is the Originator of a call session, this element MUST be present and set to “true”. It MUST be either absent or set to “false” otherwise. Default: “false”
status	ParticipantStatus	Yes	Connection status of the Participant Set by the server SHALL NOT be present in request bodies during resource creation
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids creating a resource twice for the same Participant in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named participantInformation of type ParticipantInformation is allowed in request and/or response bodies.

Regarding the clientCorrelator field, the note in section 5.2.2.2 applies.

5.2.2.17 Type: IsComposing

This type represents a message indicates to the recipient that the Sender is editing (composing) a message.

Element	Type	Optional	Description
state	xsd:string	No	Sender state, as defined in [RFC3994]. One of “idle”, “active”
lastactive	xsd:dateTime	Yes	Time of last activity, as defined in [RFC3994]

contenttype	xsd:string	Yes	Type of message being created, as defined in [RFC3994] This element contains either a MIME media type, or a combination of media type and subtype.
refresh	xsd:positiveInteger	Yes	Time interval in seconds after which the Receiver can expect an update from the Sender, as defined in [RFC3994]
(any)	any[0..unbounded]	Yes	Any element from another namespace, as defined in [RFC3994]

A root element named isComposing of type IsComposing is allowed in request bodies.

The structure of this message is aligned with [RFC3994]. Note that because the element names in this structure follow the syntax in [RFC3994], they do not conform to the naming conventions in OMA RESTful Network APIs as defined in [REST_WP].

5.2.2.18 Type: SubscriptionCancellationNotification

A type containing the subscription cancellation notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	CallbackData if passed by the application in the receiptRequest element during the associated subscription operation. See [REST_NetAPI_Common] for details.
reason	common:ServiceError	Yes	Reason notification is being discontinued. SHOULD be present if the reason is different from a regular expiry of the subscription.
link	common:Link[1..unbounded]	No	Link to other resources that are in relationship with the resource. There MUST be a link to the subscription that is cancelled.

A root element named subscriptionCancellationNotification of type SubscriptionCancellationNotification is allowed in request and/or response bodies.

5.2.3 Enumerations

The subsections of this section define the enumerations used in the Chat API.

5.2.3.1 Enumeration: ParticipantStatus

This enumeration defines the possible values for chat Participant status. The two values “Connected”, “Disconnected” are defined based on [SIMPLE_IM] chapter 7.2.2.10, plus an indication of a “pending” status i.e. “Invited”:

Enumeration	Description
Invited	Participant was invited to the session.
Connected	Participant is connected to the session.
Disconnected	Participant is disconnected from the session.

5.2.3.2 Enumeration: EventType

This enumeration is defines the types of events. It is used in notifications.

Enumeration	Description
SessionCancelled	The Originator has cancelled the chat session during the invite phase (in Confirmed 1-1 Chat and in group chat).
SessionEnded	The session has ended (in Confirmed 1-1 Chat and in group chat).
Declined	The Participant has declined the chat session invite (only in Confirmed 1-1 Chat).
Accepted	The Participant has accepted the chat invite (only in Confirmed 1-1 Chat).
Timeout	The session invitation to the Participant has timed out (only in Confirmed 1-1 Chat).
Unreachable	The Participant could not be reached or is unknown (only in Confirmed 1-1 Chat).

5.2.3.3 Enumeration: MessageStatus

This enumeration defines the possible values for the message status.

Enumeration	Description
Sent	Message was sent to the first hop in the network and has not yet reached the recipient. Initial status of a message, not used in PUT requests from the client.
Delivered	Message was delivered to the client. Maps to “delivered” according to [RFC5438] or to success reports (“Success-Report=yes”) according to [RFC4975]. Only used in notifications from the server, but not in PUT requests from the client.
Displayed	Message was displayed by the client. Maps to “displayed” according to [RFC5438].
Failed	Message was not delivered to the client. Only used in notifications from the server, but not in PUT requests from the client. Maps to failure reports (“Failure-Report=yes”) according to [RFC4975], or any other means to detect failure available to the implementation.

5.2.4 Values of the Link “rel” attribute

The “rel” attribute of the Link element is a free string set by the server implementation, to indicate a relationship between the current resource and an external resource. The following are possible strings (list is non-exhaustive, and can be extended):

- ChatSubscriptionList
- ChatNotificationSubscription
- ChatMessage
- MessageStatusReport
- ParticipantSessionStatus
- ChatSessionInformation
- GroupChatSessionInformation
- ParticipantList
- ParticipantInformation

These values indicate the kind of resource that the link points to.

5.3 Sequence Diagrams

The following sub-sections describe the resources, methods and steps involved in typical scenarios.

The sequence diagrams depict the special case where all Participants use the service via the API, because this allows illustrating the complete functionality of the API. Note that other scenarios are assumed to be more common, such as those where some Participants are connected to the service via the API, while others are using the native underlying enablers.

In a sequence diagram, a step which involves delivering a notification is labeled with “POST or NOTIFY”, where “POST” refers to delivery via the HTTP POST method, and “NOTIFY” refers to delivery using the Notification Channel [REST_NetAPI_NotificationChannel].

5.3.1 Subscription to chat notifications

The figure below shows a scenario for an application subscribing to chat notifications.

The notification URL passed by the client during the subscription step can be a Client-side Notification URL, or a Server-side Notification URL. Refer to [REST_NetAPI_NotificationChannel] for sequence flows illustrating the creation of a Notification Channel and obtaining a Server-side Notification URL on the server-side, and its use by the client via Long Polling. The resources:

- To subscribe to chat notifications, create a new resource under **http://{serverRoot}/{apiVersion}/chat/{userId}/subscriptions**
- To cancel subscription to chat notifications delete the resource under **http://{serverRoot}/{apiVersion}/chat/{userId}/subscriptions/{subscriptionId}**

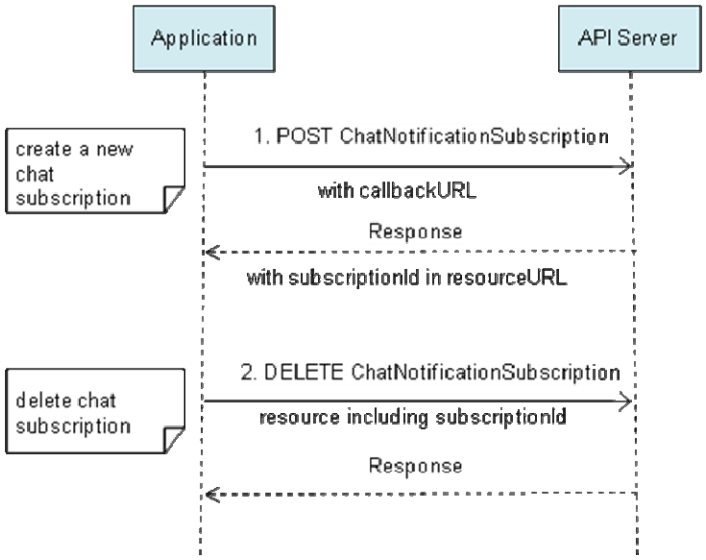

Figure 2 Subscribe to and unsubscribe from chat notifications

Outline of the flows:

1. An application subscribes to chat notifications using the POST method to submit the ChatNotificationSubscription data structure to the resource containing all subscriptions and receives the result resource URL containing the subscriptionId.
2. The application stops receiving notifications (including chat messages) using DELETE with a resource URL containing the subscriptionId.

5.3.2 Normal flow of an Ad-hoc 1-1 Chat

The figure below shows a scenario for an application to send, receive and confirm delivery of a chat message. In case of 1-1 chats, the application can immediately send the message to the desired Participant. The conversation does not need to be explicitly cancelled but automatically ends if one of the Participants stops sending chat messages for a certain time interval controlled by service provider policies.

See section 5.3.3 for an alternative, the Confirmed 1-1 Chat approach.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To send a 1-1 chat message, create a new resource at **http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/adhoc/messages**
- (The chat message is received in a notification)
- To confirm successful message reception in a 1-1 chat, update the resource at **http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/adhoc/messages/{messageId}/status**

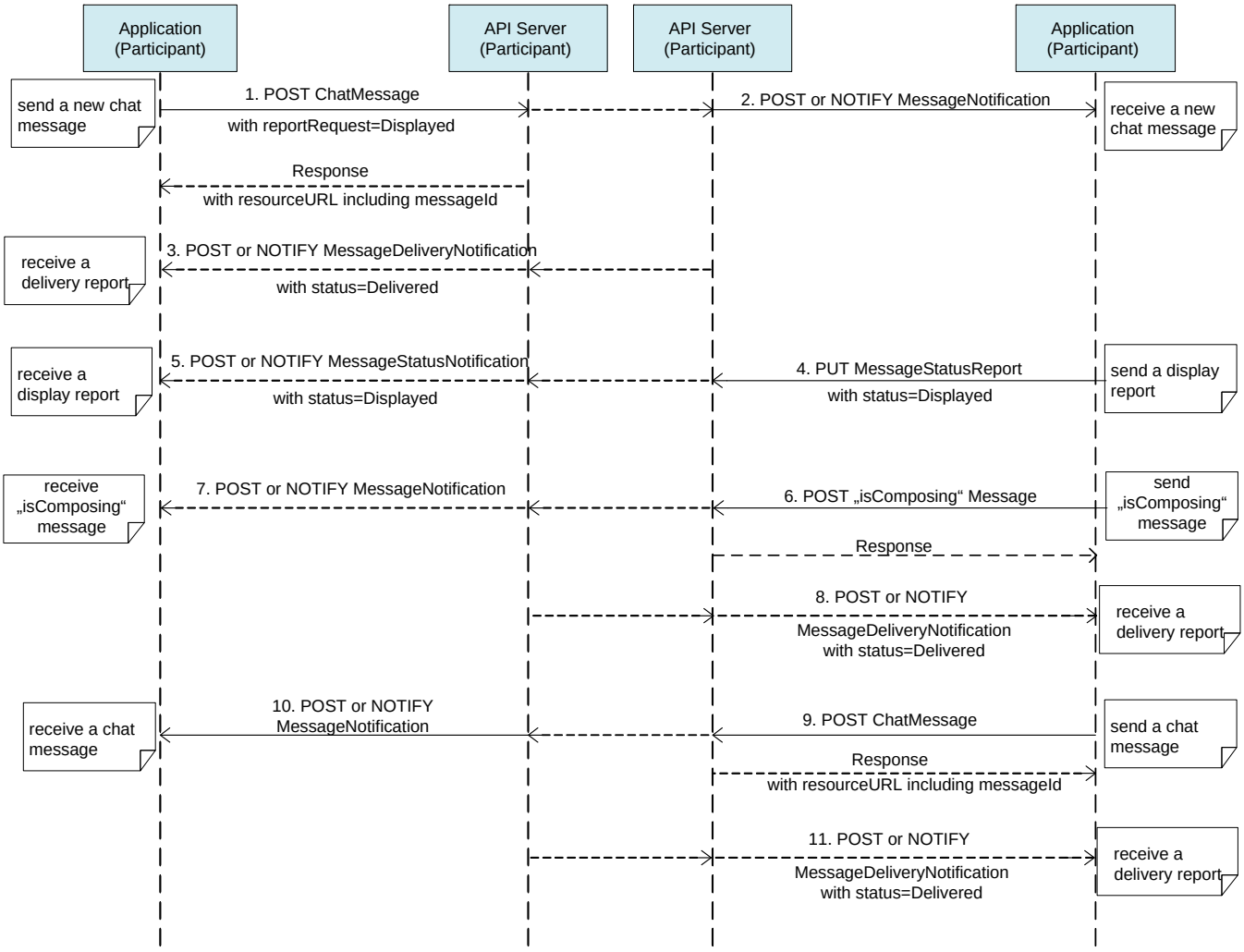

Figure 3 Normal flow of an Ad-hoc 1-1 Chat

Outline of the flows:

1. The originating application sends a chat message using the POST method to submit the ChatMessage data structure to the resource representing a container for all messages. Thereby the creation of a new chat message resource is triggered and the application receives the resulting resource URL containing the messageId.
2. The terminating application receives a chat message as a notification.
3. If the originating application has requested to receive a delivery report for the message by setting the according value in the ChatMessage data structure, a MessageStatusNotification is generated by the originator's API server (based on underlying protocol signalling in the network) and sent to the originating application.
4. If the received message contains an indication that display confirmation is requested, the terminating application confirms message display using the PUT method to submit the MessageStatusReport data structure to the resource containing the message status. Thereby the creation of a message status report is triggered.
5. If the originating application has requested to receive a display report for the message by setting the according value in the ChatMessage data structure, a MessageStatusNotification will be sent to the originating application if the message was displayed to the user.

6. As the user of the terminating application composes a chat message, the terminating application indicates that fact by sending an "isComposing" message (which is a specific chat message) to the originator's application.
7. The originating application receives the "isComposing" message as a notification.
8. To confirm delivery of the "isComposing" message, a MessageStatusNotification is generated by the Terminating Participant's API server (based on underlying protocol signalling in the network) and sent to the terminating application.
9. The terminating application replies to the Originator's message by sending a chat message using the POST method to submit the ChatMessage data structure to the resource representing a container for all messages. Thereby the creation of a new chat message resource is triggered and the application receives the resulting resource URL containing the messageId.
10. The originating application receives a chat message as a notification.
11. To confirm delivery of the chat message, a MessageStatusNotification is generated by the Originating Participant's API server (based on underlying protocol signalling in the network) and sent to the originating application.

5.3.3 Normal flow of a Confirmed 1-1 Chat

The figure below shows a scenario for a Confirmed 1-1 Chat.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To invite a user to a 1-1 chat session create a new resource under
  http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}
  with the ChatSessionInformation data structure.
- To accept a 1-1 chat session invitation, update the session status resource
  http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/status
  from "Invited" to "Connected" with the ParticipantSessionStatus data structure.

  (The originator receives a ChatEventNotification that the other user accepted the 1-1 chat session invitation)
- To send a 1-1 chat message create a new resource at
  http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/messages
  wth the ChatMessage data stucture.

  (The chat message is received in a notification)
- To confirm successful message display in a 1-1 chat update the resource at
  http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/messages/{messageId}/status
  with "Displayed" in the MessageStatusReport data structure
- To close a 1-1 chat session delete the resource
  http://{serverRoot}/chat/{apiVersion}/{userId}/ oneToOne/{otherUserId}/{sessionId}

  (Both users receive a ChatEventNotification that the session has ended)

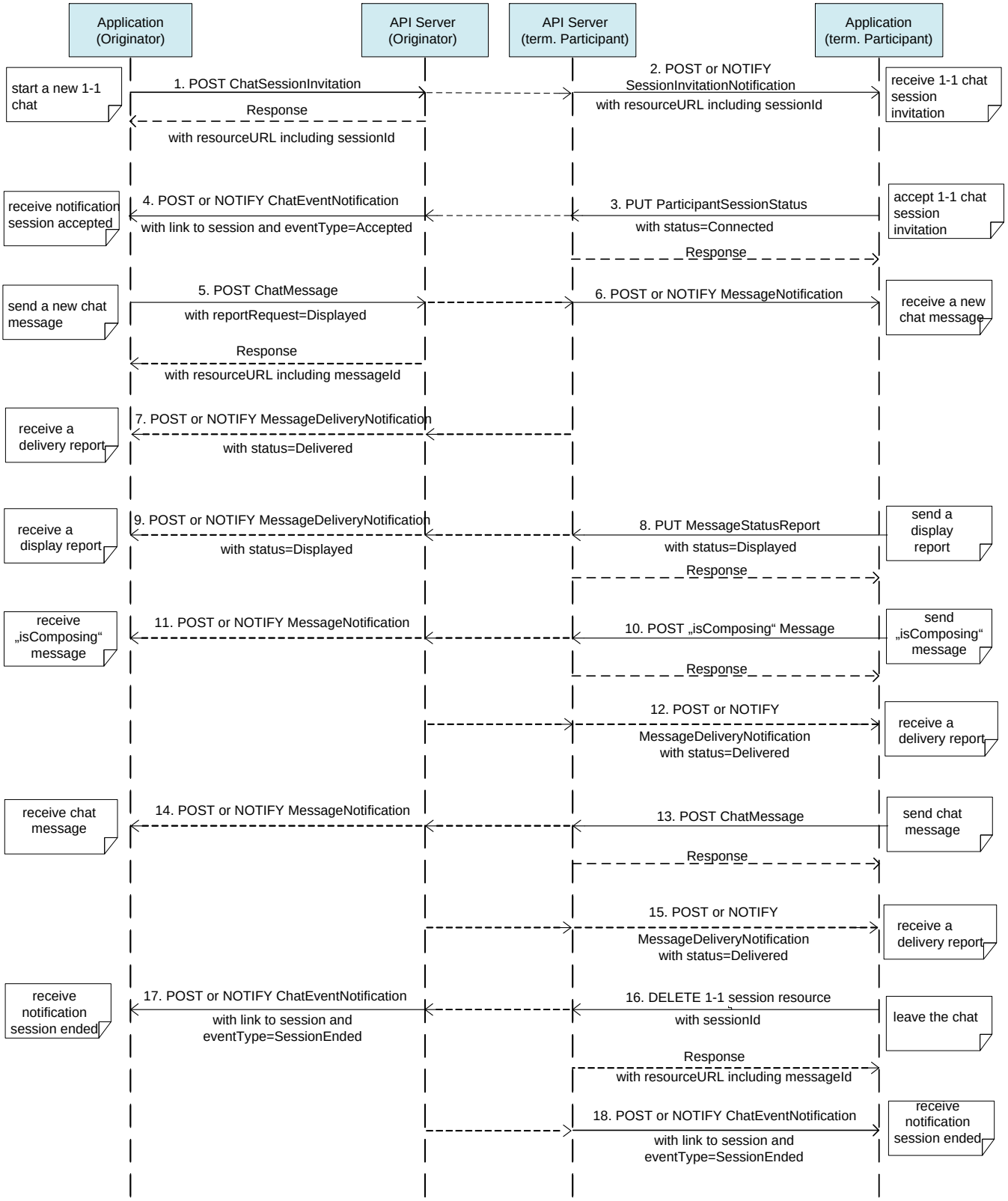

Figure 4 Normal flow of a Confirmed 1-1 Chat

Outline of the flows:

1. The originating application sends a chat session invitation using the POST method to submit the ChatSessionInformation data structure to the resource representing the chat sessions with the invited user. Thereby the creation of a new chat session resource is triggered and the application receives the resulting resource URL containing the sessionId.
2. The terminating application receives a chat session invitation as a notification.
3. The terminating application accepts the chat session invitation using the PUT method to submit the ParticipantSessionStatus data structure to the resource representing the session status, updating the status from 'Invited' to 'Connected'.
4. The originating application receives a chat event notification including a link to the resource representing the session that has been accepted.
5. The originating application sends a chat message using the POST method to submit the ChatMessage data structure to the resource representing a container for all messages in the current chat session. Thereby the creation of a new chat message resource is triggered and the application receives the resulting resource URL containing the messageId.
6. The terminating application receives the chat message as a notification.
7. If the originating application has requested to receive a delivery report for the message by setting the according value in the ChatMessage data structure, a MessageStatusNotification is generated by the originator's API server (based on underlying protocol signalling in the network) and sent to the originating application.
8. If the received message contains an indication that display confirmation is requested, the terminating application confirms message display using the PUT method to submit the MessageStatusReport data structure to the resource containing the message status. Thereby the creation of a MessageStatusReport notification is triggered.
9. If the originating application has requested to receive a success report for the display of the message by setting the according value in the ChatMessage data structure, a MessageStatusNotification will be sent to the originating application if the message was displayed.
10. As the user of the terminating application composes a chat message, the terminating application indicates that fact by sending an "isComposing" message (which is a specific chat message) to the originator's application.
11. The originating application receives the "isComposing" message as a notification.
12. To confirm delivery of the "isComposing" message, a MessageStatusNotification is generated by the terminating Participant's API server (based on underlying protocol signalling in the network) and sent to the terminating application.
13. The terminating application sends a chat message using the POST method to submit the ChatMessage data structure to the resource representing a container for all messages. Thereby the creation of a new chat message resource is triggered and the application receives the resulting resource URL containing the messageId.
14. The originating application receives a chat message as a notification.
15. If the terminating application has requested to receive a delivery report for the message by setting the according value in the ChatMessage data structure, a MessageStatusNotification is generated by the terminating participant's API server (based on underlying protocol signalling in the network) and sent to the terminating application.

    (Note that more messages may be exchanged in the session)
16. The terminating application closes the chat session using the DELETE method to the resource representing the chat session. Thereby a notification is triggered that the session has ended. (Note that the originating application can also close the session).
17. The originating application receives a chat event notification that the session provided in the link has ended.
18. The terminating application receives a chat event notification that the session provided in the link has ended.

5.3.4 Declining an invitation to a Confirmed 1-1 Chat

The figure below shows a scenario for declining a Confirmed 1-1 Chat invitation.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To decline an invitation to a Confirmed 1-1 Chat session delete the resource **http://{serverRoot}/chat/{apiVersion}/{userId}/ oneToOne/{otherUserId}/{sessionId}**

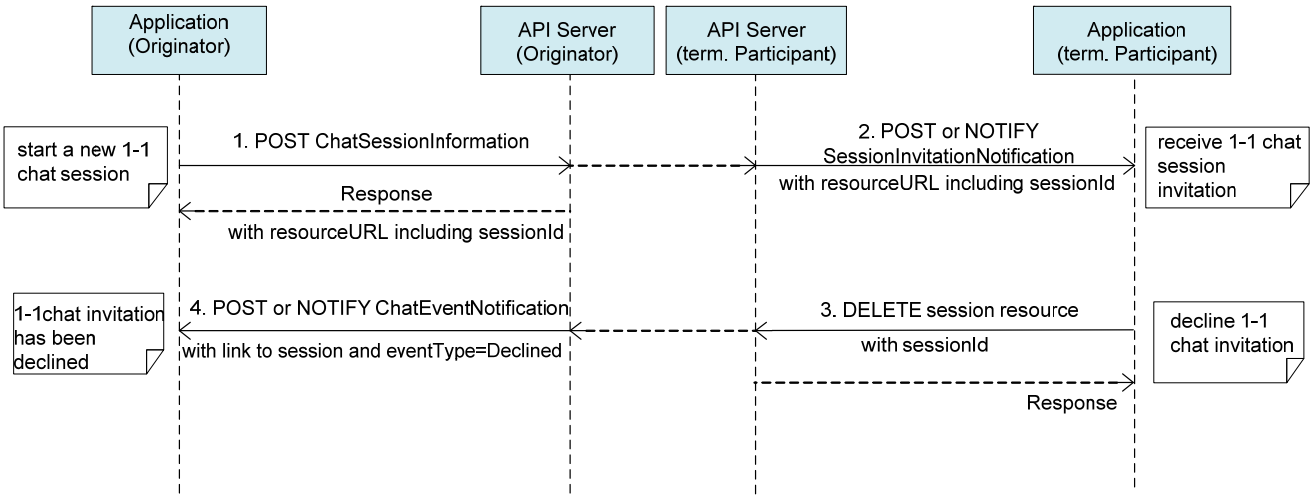

Figure 5 Declining an invitation to a Confirmed 1-1 Chat

Outline of the flows:

1. The originating application sends an invitation to a Confirmed 1-1 Chat session invitation using the POST method to submit the ChatSessionInformation data structure to the resource representing the chat sessions with invited user. Thereby the creation of a new chat session resource is triggered and the application receives the resulting resource URL containing the sessionId.
2. The terminating application receives a chat session invitation as a notification.
3. The terminating application declines the chat session invitation using the DELETE method to the resource representing the chat session. Thereby a notification is triggered that the session has been declined.
4. A chat event notification is sent to the originator application that the session has been declined.

5.3.5 Cancelling an invitation to a Confirmed 1-1 Chat

The figure below shows a scenario for an originator cancelling a in invitation to a Confirmed 1-1 Chat.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To cancel a 1-1 chat session invitation delete the resource **http://{serverRoot}/chat/{apiVersion}/{userId}/ oneToOne/{otherUserId}/{sessionId}**

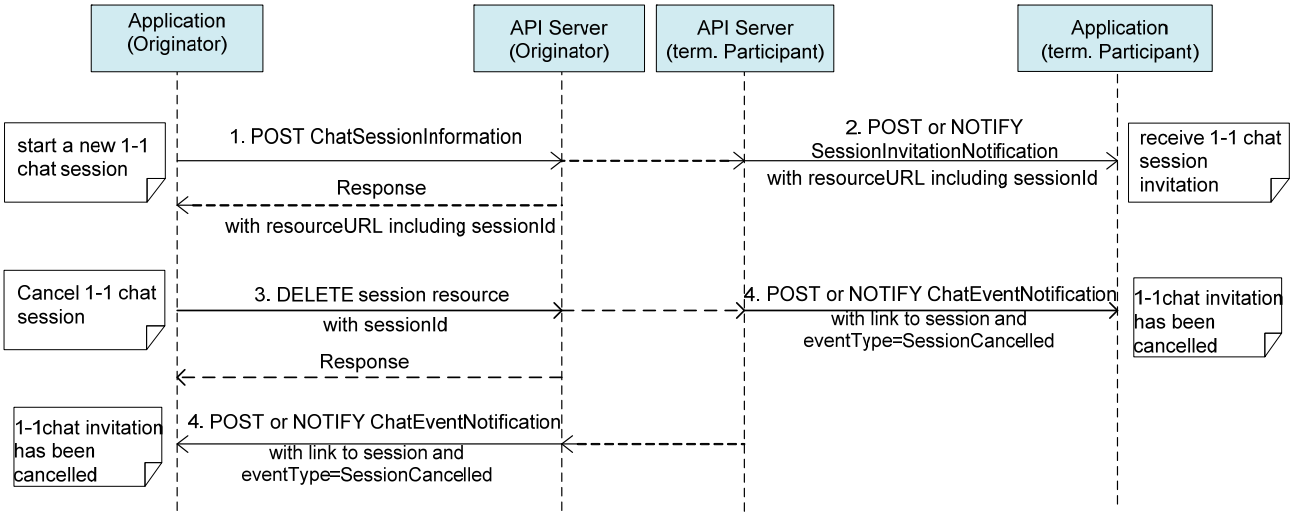

Figure 6 Cancelling an invitation to a Confirmed 1-1 Chat

Outline of the flows:

1. The originating application sends a 1-1 chat session invitation using the POST method to submit the ChatSessionInformation data structure to the resource representing the invited user. Thereby the creation of a new chat session resource is triggered and the application receives the resulting resource URL containing the sessionId.
2. The terminating application receives a chat session invitation as a notification.
3. The terminating application cancels the chat session invitation using the DELETE method to the resource representing the chat session. Thereby a notification is triggered that the session has been cancelled.
4. A chat event notification is send to the originating and terminating applications that the session has been cancelled.

5.3.6 Normal flow of a group chat

The figure below shows a scenario for a normal group chat.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To start a group chat session create a new resource under
  http://{serverRoot}/chat/{apiVersion}/{userId}/group
  with the GroupChatSessionInformation data structure.
- To accept a group chat session invitation update the Participant status resource
  http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}/status
- To send a group chat message create a new resource at
  http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/messages
- To invite additional Participants to the existing group chat session update the resource
  http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants
- To leave a group chat session delete the resource
  http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}
- To re-join a group chat session POST the ParticipantInformation to
  http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants

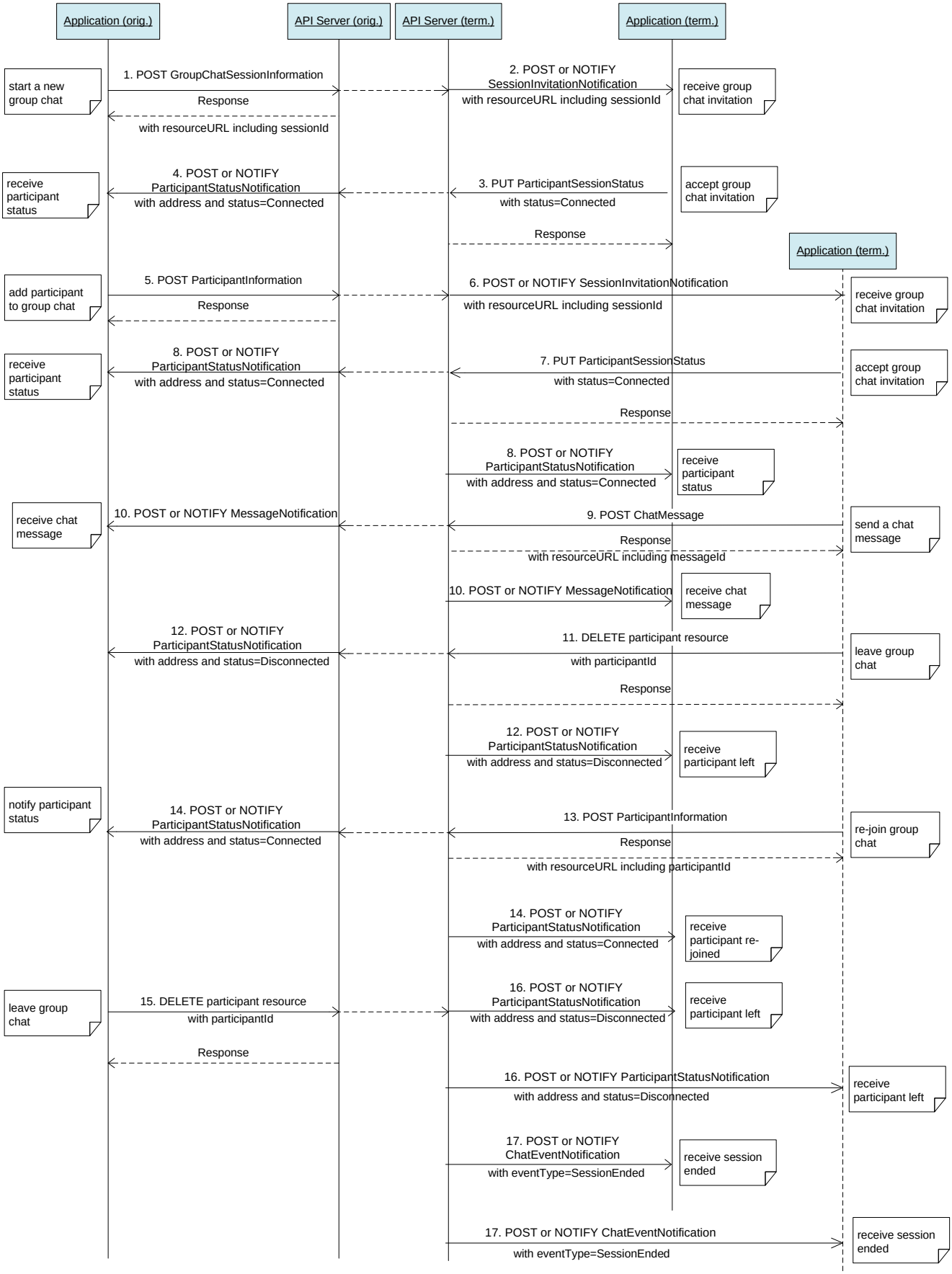

Figure 7 Normal flow of group chat

Outline of the flow:

1. The originating application starts a group chat session using the POST method to submit to the resource containing all group chat sessions the GroupChatSessionInformation data structure with the list of invited Participants. Thereby the creation of a new chat session resource is triggered and the application receives the resulting resource URL containing the sessionId.
2. The terminating application receives a group chat SessionInvitationNotification as a notification.
3. The terminating application accepts the group chat invitation using the PUT method to submit the ParticipantSessionStatus data structure to the resource containing the Participant status. The status MUST be set to "Connected".
4. The originating application receives a ParticipantStatusNotification data structure with status information of the Participant(s).
5. The originating application adds another Participant to the group chat session using the POST method to update the ParticipantInformation data structure under the resource that contains all Participants. Thereby the originating API server triggers a new SessionInvitationNotification to the newly added Participant.
6. The application of the new terminating Participant receives a group chat SessionInvitationNotification.
7. The terminating application accepts the group chat invitation using the PUT method to submit the ParticipantSessionStatus data structure to the resource containing the Participant status. The status MUST be set to "Connected".
8. All applications connected to the group chat receive a ParticipantStatusNotification data structures with status information of the Participant.
9. The application sends a chat message using the POST method to submit the ChatMessage data structure to the resource representing a container for all messages. Thereby the creation of a new chat message resource is triggered and the application receives the resulting resource URL containing the messageId.
10. All applications connected to the chat session receive the chat message as a notification.
11. An application leaves a group chat session using the DELETE method on the "participants" resource including the participantId. The Participant is thereby deleted from the Participants list while the session still exists (as this is a group chat session.)
12. A ParticipantStatusNotification is created by the API server to inform all other Participants that a user has left.
13. An application re-joins a group chat session using the POST method to submit the ParticipantInformation data structure to the resource containing the Participants. The status MUST be set to "Connected". The application receives a resource URL containing a new participantId.
14. A ParticipantStatusNotification is created by the API server to inform all Participants that a user has joined.
15. The Originator's application leaves a group chat session using the DELETE method on the "participants" resource including the participantId. The Originator is thereby deleted from the Participants list while the session may stay alive for remaining Participants or may end (depending on service provider policies, see step 17.)
16. A ParticipantStatusNotification is created by the API server to inform all other Participants that a user has left.
17. Depending on service provider policies the session may end when the Originator has left the chat, which triggers a ChatEventNotification (SessionEnded) to inform the remaining Participants that the group chat session has ended.

At minimum, a group chat session consists of the steps 1, 2, 3, 4, 9, 10 and 17.

Note that a group chat session terminates according to service provider policies when all Participants have left, or when the Originator has left, or after a specific period of time (e.g. pre-defined maximum session duration, inactivity, etc.).

5.3.7 Declining a group chat session invitation

The figure below shows how to decline a group chat session invitation.

This flow only applies if the server has not automatically accepted the invitation as stated in section 5.3.6 step 3 and 7.

The precondition for this flow to work is that the client has subscribed to chat event notifications, see section 5.3.1.

The resources:

- To decline a group chat session invitation delete the Participant resource in the chat session **http://{serverRoot}/{apiVersion}/chat/{userId}/group/{sessionId}/participants/{participantId}**

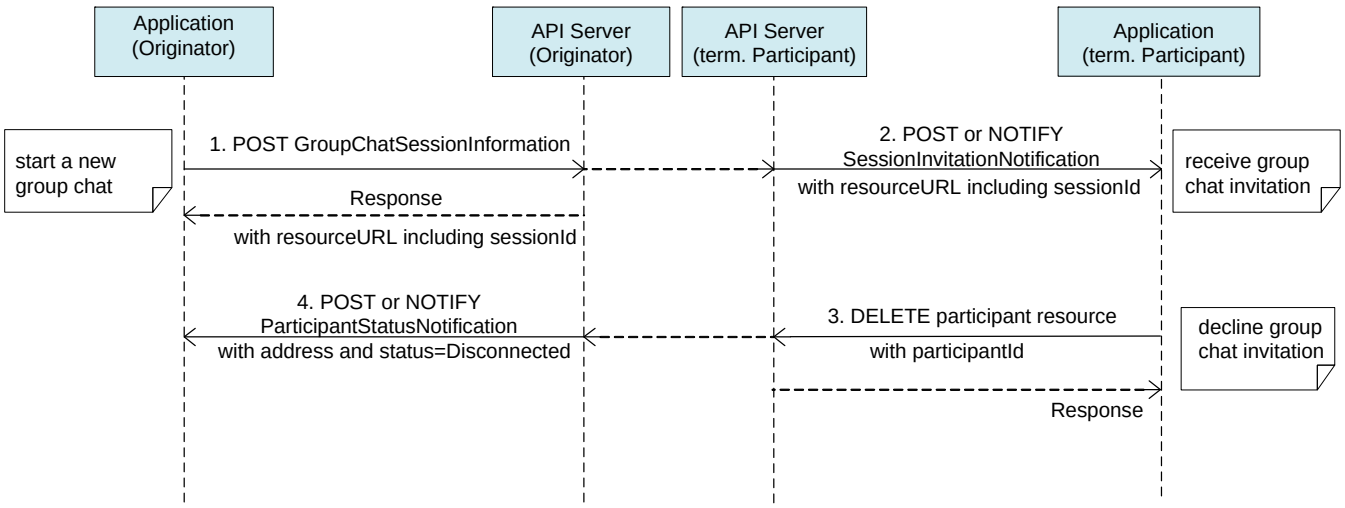

Figure 8: Declining a group chat invitation

Outline of the flow:

1. The originating application starts a group chat session using the POST method to submit to the resource containing all group chat sessions the GroupChatSessionInformation data structure with the list of invited Participants. Thereby the creation of a new chat session resource is triggered and the application receives the resulting resource URL containing the sessionId.
2. The terminating application receives a group chat SessionInvitationNotification as a notification.
3. The terminating application declines the group chat session invitation using the DELETE method on the “participants” resource including the participantId. The Participant is thereby deleted from the Participants list of the session, while the session still exists (as this is a group chat session.)
4. A ParticipantStatusNotification is created by the API server to inform all other Participants that a user has left.

6. Detailed specification of the resources

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML, JSON, application/x-www-form-urlencoded):

- Reserved characters in URL variables (parts of a URL denoted below by a name in curly brackets) MUST be percent-encoded according to [RFC3986]. Note that this always applies, no matter whether the URL is used as a Request URL or inside the representation of a resource (such as in “resourceURL” and “link” elements).
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an MSISDN, it MUST be defined as a global number according to [RFC3966] (e.g. tel:+19585550100). The use of characters other than digits and the leading “+” sign SHOULD be avoided in order to ensure uniqueness of the resource URL. This applies regardless of whether the user identifier appears in a URL variable or in a parameter in the body of an HTTP message.
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of a SIP URI, it MUST be defined according to [RFC3261].
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an Anonymous Customer Reference (ACR), it MUST be defined according to [IETF_ACR_draft], i.e. it MUST include the protocol prefix 'acr:' followed by the ACR.
    - The ACR ‘Authorization’ is a supported reserved keyword, and MUST NOT be assigned as an ACR to any particular end user. See G.1.2 for details regarding the use of this reserved keyword.
- For requests and responses that have a body, the following applies: in the requests received, the server SHALL support JSON and XML encoding of the parameters in the body, and MAY support application/x-www-form-urlencoded parameters in the body. The Server SHALL return either JSON or XML encoded parameters in the response body, according to the result of the content type negotiation as specified in [REST_NetAPI_Common]. In notifications to the Client, the server SHALL use either XML or JSON encoding, depending on which format the client has specified in the related subscription. The generation and handling of the JSON representations SHALL follow the rules for JSON encoding in HTTP Requests/Responses as specified in [REST_ NetAPI_Common].

6.1 Resource: All subscriptions to chat event notifications

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/subscriptions

This resource is used to manage subscriptions to chat event notifications. Note that there is one subscription per client instance.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.1.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description

serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.1.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.1.3 GET

This operation is used for reading the list of active chat notification subscriptions.

6.1.3.1 Example: Reading all active chat notification subscriptions (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatSubscriptionList xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <chatNotificationSubscription>
        <callbackReference>
            <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
            <callbackData>abcd</callbackData>
        </callbackReference>
        <duration>7037</duration>
        <clientCorrelator>12345</clientCorrelator>
        <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
    </chatNotificationSubscription>
    <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions</resourceURL>
</chat:chatSubscriptionList>
```

6.1.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC 2616].

6.1.5 POST

This operation is used to create a new subscription for chat notifications.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.1.5.1 Example 1: Creating a new subscription to chat notifications, response with copy of created resource (Informative)

6.1.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <callbackReference>
        <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</chat:chatNotificationSubscription>
```

6.1.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackReference>
        <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
  </callbackReference>
  <duration>7200</duration>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</chat:chatNotificationSubscription>
```

6.1.5.2 Example 2: Creating a new subscription to chat notifications, response with location of created resource (Informative)

6.1.5.2.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <callbackReference>
        <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</chat:chatNotificationSubscription>
```

6.1.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</common:resourceReference>
```

6.1.5.3 Example 3: Creating a new subscription to chat notifications, requiring support of Confirmed 1-1 Chats which the server does not provide (Informative)

6.1.5.3.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <callbackReference>
        <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
```

```
    </callbackReference>
    <confirmedChatSupported>true</confirmedChatSupported>
    <adhocChatSupported>false</adhocChatSupported>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</chat:chatNotificationSubscription>
```

6.1.5.3.2 Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
      <policyException>
         <messageId>POL1013</messageId>
         <text>Confirmed 1-1 chats are not supported.</text>
      </policyException>
</common:requestError>
```

6.1.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC 2616].

6.2 Resource: Individual subscription to chat event notifications

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/subscriptions/{subscriptionId}

This resource represents an individual subscription to chat notifications.

6.2.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123

subscriptionId	Identifier of the subscription

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.2.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.2.3 GET

This operation is used for reading an individual subscription.

6.2.3.1 Example: Reading an individual subscription (Informative)

This example shows also an alternative way to indicate desired content type in response from the server, by using URL query parameter "?resFormat" which is described in [REST_NetAPI_Common].

6.2.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=XML HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <callbackReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
        <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
    <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</chat:chatNotificationSubscription>
```

6.2.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.2.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.2.6 DELETE

This operation is used to cancel a subscription and to stop corresponding notifications.

6.2.6.1 Example: Cancelling a subscription (Informative)

6.2.6.1.1 Request

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

6.3 Resource: All 1-1 chat sessions between two users

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/

This resource contains information about all 1-1 chat sessions between two particular users.

6.3.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI.
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.3.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.3.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.3.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.3.5 POST

This operation is used to create a 1-1 chat session with the user represented by {userId} as Originator and the one represented by {otherUserId} as Terminating Participant.

For Ad-hoc 1-1 Chats (i.e. those using the reserved value "adhoc" for {sessionId}), this step is not necessary.

6.3.5.1 Example: Creating a 1-1 chat session (Informative)

6.3.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: example.com
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatSessionInformation xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <subject>Hi there!</subject>
   <originatorAddress>tel:+19585550100</originatorAddress>
   <originatorName>Alice</originatorName>
   <tParticipantAddress>tel:+19585550101</tParticipantAddress>
   <tParticipantName>Bob</tParticipantName>
</chat:chatSessionInformation>
```

6.3.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatSessionInformation xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <subject>Hi there!</subject>
```

```
   <originatorAddress>tel:+19585550100</originatorAddress>
   <originatorName>Alice</originatorName>
   <tParticipantAddress>tel:+19585550101</tParticipantAddress>
   <tParticipantName>Bob</tParticipantName>
   <status>Invited</status>
   <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001
   </resourceURL>
</chat:chatSessionInformation>
```

6.3.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.4 Resource: Individual 1-1 chat session

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/

This resource represents a 1-1 chat session.

A 1-1 chat session MAY be extended to a group chat session as described in section 6.6. These are represented using different resources because the feature sets of both types of sessions are different. In case a 1-1 session has been successfully extended into a group chat session, the 1-1 session is closed. For a certain period of time after extending the session, it is RECOMMENDED to redirect all accesses to a 1-1 session resource or its offspring resources to the resource representing the corresponding group chat session. Section 6.4.3.2 provides an example for such redirection.

6.4.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI.
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session In Confirmed 1-1 Chat mode, this identifier is populated by the server during resource creation. In Ad-hoc 1-1 Chat mode, this identifier SHALL be set to the reserved word "adhoc",

	which SHALL NOT be used for other purposes in this resource URL variable.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.4.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7:

6.4.3 GET

This operation is used to retrieve chat session information.

6.4.3.1 Example 1: Retrieving chat session information of a 1-1 session (Informative)

6.4.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatSessionInformation xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <subject>Hi there!</subject>
   <originatorAddress>tel:+19585550100</originatorAddress>
   <originatorName>Alice</originatorName>
   <tParticipantAddress>tel:+19585550101</tParticipantAddress>
   <tParticipantName>Bob</tParticipantName>
   <status>Invited</status>
   <resourceURL>
http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001
   </resourceURL>
</chat:chatSessionInformation>
```

6.4.3.2 Example 2: Retrieving chat session information of a 1-1 session that was previously extended to a group chat session (Informative)

6.4.3.2.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/xml
```

```
Host: example.com
```

6.4.3.2.2 Response

```
HTTP/1.1 303 See Other
Content-Type: application/xml
Location: /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</common:resourceReference>
```

6.4.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.4.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.4.6 DELETE

This operation ends the chat session.

It is used in the following contexts:

- by the Originator to cancel a pending invitation before the Terminating Participant has accepted the invitation, which will cause the session to end
- by the Terminating Participant to decline an invitation to a chat session, which will cause the session to end
- by any Participant to terminate the chat session.

6.4.6.1 Example: Terminating a 1-1 chat session, or declining an invitation (Informative)

6.4.6.1.1 Request

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.5 Resource: 1-1 chat session status

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/status

This resource represents the status of the session and is used for accepting a 1-1 chat invitation, by means of updating the status.

6.5.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session In Confirmed 1-1 Chat mode, this identifier is populated by the server during resource creation. In Ad-hoc 1-1 Chat mode, this identifier SHALL be set to the reserved word “adhoc”, which SHALL NOT be used for other purposes in this resource URL variable.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.5.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7:

6.5.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: PUT’ field in the response as per section 14.7 of [RFC 2616].

6.5.4 PUT

This operation is used is used for accepting a 1-1 chat invitation, by means of updating the status.

6.5.4.1 Example 1: Accepting a 1-1 chat invitation (Informative)

6.5.4.1.1 Request

```
PUT /exampleAPI/chat/v1/tel%3A%2B19585550101/oneToOne/{otherUserId}/sess001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantSessionStatus xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <status>Connected</status>
</chat:participantSessionStatus>
```

6.5.4.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2011 17:51:59 GMT
```

6.5.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: PUT’ field in the response as per section 14.7 of [RFC 2616].

6.5.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: PUT’ field in the response as per section 14.7 of [RFC 2616].

6.6 Resource: Extend 1-1 chat to a group chat session

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/extend

This resource is used to extend a 1-1 chat to a group chat session.

6.6.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use. The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123.
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123.
sessionId	Identifier of the chat session In Confirmed 1-1 Chat mode, this identifier is populated by the server during resource creation. In Ad-hoc 1-1 Chat mode, this identifier SHALL be set to the reserved word “adhoc”, which SHALL NOT be used for other purposes in this resource URL variable.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.6.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.6.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.6.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.6.5 POST

This operation is used to extend a 1-1 chat to a group chat session. The list in the body of the request represents those users that are to be added to the session.

In case of successful operation, “303 See Other” SHALL be returned, providing a Location header and a resourceReference root element with the location representing the new group chat session in which the Originator is already a Participant. All Participants given in the body of the HTTP request are invited to the group chat session.

On behalf of the Terminating Participant in the original 1-1 session, the API server SHALL end the original 1-1 chat session once the Terminating Participant in the original 1-1 session has accepted or declined the invitation to the group chat, or once that invitation has timed out.

6.6.5.1 Example: Extending a Confirmed 1-1 Chat to a group chat session (Informative)

6.6.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/extend HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantList xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
      <participant>
       <address>tel:+B19585550102</address>
       <name>Ted</name>
       <clientCorrelator>ABCDE</clientCorrelator>
   </participant>
</chat:participantList >
```

6.6.5.1.2 Response

```
HTTP/1.1 303 See Other
Content-Type: application/xml
Location: /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
   <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</common:resourceReference>
```

6.6.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.7 Resource: Chat messages in a 1-1 chat

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/messages

This resource represents all chat messages in a chat session. In the current version of the specification, it is a "send-only" resource (i.e. chat messages cannot be read back).

6.7.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session In Confirmed 1-1 Chat mode, this identifier is populated by the server during resource creation. In Ad-hoc 1-1 Chat mode, this identifier SHALL be set to the reserved word "adhoc", which SHALL NOT be used for other purposes in this resource URL variable.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.7.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.7.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.7.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.7.5 POST

This operation is used to create a chat message. This method MUST return either a resourceReference root element or a chatMessage root element, where using the first option is RECOMMENDED.

6.7.5.1 Example 1: Creating a chat message, using tel URI and returning the location of the created resource (Informative)

6.7.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <text>How are you?</text>
  <reportRequest>Displayed</reportRequest>
</chat:chatMessage>
```

6.7.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location:
http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
 <resourceURL>
  http://example.com/exampleAPI/chat/v1/ tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001
 </resourceURL>
</common:resourceReference>
```

Note that alternatively, a copy of the created resource can be returned, as illustrated in section 6.7.5.2.

6.7.5.2 Example 2: Creating a chat message, using ACR and returning a copy of the created resource (Informative)

6.7.5.2.1 Request

```
POST /exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/adhoc/messages HTTP/1.1
```

```
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <text>How are you?</text>
  <reportRequest>Displayed</reportRequest>
</chat:chatMessage>
```

6.7.5.2.2 Response

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/adhoc/messages/msg001
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <text>How are you?</text>
  <reportRequest>Displayed</reportRequest>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/messages/adhoc/msg001
  </resourceURL>
</chat:chatMessage>
```

Note that alternatively, the location of the created resource can be returned, as illustrated in section 6.7.5.1.

6.7.5.3 Example 3: Creating an “isComposing” message and returning the location of the created resource (Informative)

6.7.5.3.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:isComposing xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <state>active</state>
  <contenttype>text/plain</contenttype>
  <refresh>90</refresh>
</chat:isComposing>
```

6.7.5.3.2 Response

```
HTTP/1.1 201 Created
```

```
Content-Type: application/xml
Location:
http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg002
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/ tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg002
  </resourceURL>
</common:resourceReference>
```

Note that alternatively, a copy of the created resource can be returned, as illustrated in section 6.7.5.2.

6.7.5.4 Example 4: Creating a chat message during session set-up in Confirmed 1-1 Chat mode (Informative)

This example illustrates the case of trying to send a chat message between creating the chat session and receiving the acceptance message from the terminating participant. In case the Chat server does not buffer such messages, an exception is returned as follows.

6.7.5.4.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/messages HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <text>How are you?</text>
    <reportRequest>Displayed</reportRequest>
</chat:chatMessage>
```

6.7.5.4.2 Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
    <link rel="ChatMessage"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/messages/msg002
"/>
    <policyException>
        <messageId>POL1012</messageId>
        <text>Messages during session setup not supported.</text>
    </policyException>
</common:requestError>
```

6.7.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.8 Resource: Individual message status in a 1-1 chat

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne/{otherUserId}/{sessionId}/messages/{messageId}/status

This resource represents the status of a message.

Note: The duration for which the Server stores information about a chat message is controlled by service provider policies.

6.8.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI.
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
otherUserId	Identifier of the user who acts as chat partner Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session In Confirmed 1-1 Chat mode, this identifier is populated by the server during resource creation. In Ad-hoc 1-1 Chat mode, this identifier SHALL be set to the reserved word "adhoc", which SHALL NOT be used for other purposes in this resource URL variable.
messageId	Identifier of the message

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.8.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.8.3 GET

This operation is used for reading the status of an individual message.

6.8.3.1 Example: Reading the status of an individual message (Informative)

6.8.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001/status HTTP/1.1
Accept: application/xml
Host: example.com
```

6.8.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:messageStatusReport xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <status>Displayed</status>
</chat:messageStatusReport>
```

6.8.4 PUT

This operation is used for reporting the “Displayed” status of a message. The client SHALL execute this method if a received message indicates that a “Displayed” status report is requested, by including the element ‘reportRequest’ in the message.

Note that the “Delivered” status report is generated by the API Server by procedures of the underlying protocol layers which are out of scope of this specification.

6.8.4.1 Example: Reporting the status of a chat message (Informative)

6.8.4.1.1 Request

```
PUT /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:messageStatusReport xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <status>Displayed</status>
</chat:messageStatusReport>
```

6.8.4.1.2 Response

HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT

6.8.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC 2616].

6.8.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT' field in the response as per section 14.7 of [RFC 2616].

6.9 Resource: All group chat sessions

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group

This resource represents the active group chat sessions for a particular user.

6.9.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.9.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.9.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.9.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.9.5 POST

This operation is used to create a new group chat session.

6.9.5.1 Example: Creating a new group chat session (Informative)

6.9.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group HTTP/1.1
Content-Length: nnnn
Content-Type: application/xml
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:groupChatSessionInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <subject>Hello</subject>
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <isOriginator>true</isOriginator>
  </participant>
  <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
  </participant>
  <participant>
    <address>tel:+19585550102</address>
    <name>Ted</name>
  </participant>
  <clientCorrelator>12345</clientCorrelator>
</chat:groupChatSessionInformation>
```

6.9.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:groupChatSessionInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
 <subject>Hello</subject>
 <participant>
   <address>tel:+19585550100</address>
   <name>Alice</name>
   <isOriginator>true</isOriginator>
   <status>Connected</status>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
   </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550101</address>
   <name>Bob</name>
   <status>Invited</status>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550102</address>
   <name>Ted</name>
   <status>Invited</status>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
  </resourceURL>
 </participant>
 <clientCorrelator>12345</clientCorrelator>
 <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</chat:groupChatSessionInformation>
```

Note that alternatively, a ‘resourceReference’ root element can be returned, as illustrated in section 6.1.5.2.2.

6.9.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.10 Resource: Individual group chat session

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}

This resource represents a group chat session.

For a limited time after a Participant has left a group chat session, the API server still exposes the resource URL of a chat session to the client that has left for the purpose of re-joining. The time how long this is exposed is controlled by operator policies. In case the client is not a Participant of the chat session but the resource URL representing the session is exposed to him for possible re-joining, the response to the GET method SHALL be "204 No Content". Note that the reason for this is the fact that it is implementation-specific whether or not a disconnected Participant gets notifications about session progress from the underlying protocol layers.

6.10.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session This identifier is populated by the server during resource creation.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.10.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.10.3 GET

This operation is used to retrieve chat session information.

6.10.3.1 Example 1: Retrieving group chat session information (Informative)

6.10.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.10.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:groupChatSessionInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
 <subject>Hello</subject>
 <participant>
   <address>tel:+19585550100</address>
   <name>Alice</name>
   <isOriginator>true</isOriginator>
   <status>Connected</status>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550101</address>
   <name>Bob</name>
   <status>Invited</status>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550102</address>
   <name>Ted</name>
   <status>Invited</status>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
  </resourceURL>
 </participant>
 <clientCorrelator>12345</clientCorrelator>
 <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</chat:groupChatSessionInformation>
```

6.10.3.2 Example 2: Retrieving group chat session information when being disconnected (Informative)

This example illustrates the case that the client reads information about a group chat session on behalf of a user who is currently not participating in the session, but who is allowed to re-join according to operator policies.

6.10.3.2.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.10.3.2.2 Response

```
HTTP/1.1 204 No Content
```

```
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.10.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.10.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.10.6 DELETE

This operation is used by the Originator to terminate the group chat session.

6.10.6.1 Example: Terminating a group chat session (Informative)

6.10.6.1.1 Request

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.10.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.11 Resource: All Participants in a group chat session

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants

This resource represents the set of Participants in a group chat session.

For a limited time after a Participant has left a group chat session, the API server still exposes the resource URL of a chat session and the 'participants' node to the client that has left, for the purpose of re-joining. The time how long this is exposed is controlled by operator policies. During the time this resource is still available, the client can re-join by executing the POST method as described below. In case the client is not a Participant of the chat session but the resource URL representing the session is exposed to him for possible re-joining, the response to the GET method SHALL be "204 No Content". Note that the reason for this is the fact that it is implementation-specific whether or not a disconnected Participant gets notifications about session progress from the underlying protocol layers.

6.11.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session This identifier is populated by the server during resource creation.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.11.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.11.3 GET

This operation is used to retrieve the list of Participants in a group chat session.

6.11.3.1 Example 1: Retrieving the list of Participants in a group chat session (Informative)

This example illustrates the case that the client reads the list of Participants on behalf of a user who is currently participating in the session.

6.11.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/xml
Host: example.com
```

6.11.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

```
<?xml version="1.0" encoding="UTF-8"?>
<chat:participantList  xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <isOriginator>true</isOriginator>
    <status>Connected</status>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
   </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
    <status>Connected</status>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
   </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550102</address>
    <name>Ted</name>
    <status>Connected</status>
   <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
   </resourceURL>
  </participant>
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
   </resourceURL>
</chat:participantList>
```

6.11.3.2 Example 2: Retrieving the list of Participants in a group chat session when being disconnected (Informative)

This example illustrates the case that the client reads the list of Participants on behalf of a user who is currently not participating in the session, but who is allowed to re-join according to operator policies.

6.11.3.2.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/xml
Host: example.com
```

6.11.3.2.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.11.3.3 Example 3: Retrieving the list of Participants in a group chat session when not having access rights (Informative)

This example illustrates the case that the client reads the list of Participants on behalf of a user who is currently not participating in the session, and who is not allowed to re-join according to operator policies.

6.11.3.3.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/xml
Host: example.com
```

6.11.3.3.2 Response

```
HTTP/1.1 403 Forbidden
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="ParticipantList"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants"/>
  <policyException>
    <messageId>POL1011</messageId>
    <text>Access denied.</text>
  </policyException>
</common:requestError>
```

6.11.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC 2616].

6.11.5 POST

This operation is used to add Participants to a group chat:

- The Originator executes this method to add one or more Participants to a group chat.
- A former Participant executes this method to re-join a group chat.
- A Terminating Participant executes this method to add one or more Participants to a group chat, if operator policies allow that.

Note that for a Participant re-joining a chat session, the {participantId} resource URL variable is not guaranteed to have the same value as in the previous participation of the Participant in the session.

If the operation was successful, it returns an HTTP Status of "201 Created" in case there was just one tParticipantAddress instance passed, or "200 OK" if there were multiple instances passed. Further, in case there was just one tParticipantAddress instance passed, the response entity body contains either a "resourceReference" or a "participant" root element. In case there were multiple tParticipantAddress instances passed, the response entity body contains a "participantList" root element.

In other words, adding one Participant corresponds to the creation of a new “participant” resource in the list of Participants and is consistent with the resource creation design pattern used throughout the OMA RESTful Network APIs, whereas the addition of multiple Participants corresponds to an update operation of the list of Participants.

6.11.5.1 Example 1: Adding one Participant to a group chat, or re-joining a group chat (Informative)

This example illustrates the following three cases for which the same request syntax is being used:

- one Participant added to a group chat by the Originator
- one Participant re-joining a chat which she/he has left earlier

6.11.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
   <address>tel:+19585550103</address>
   <name>John</name>
    <clientCorrelator>12345</clientCorrelator>
</chat:participantInformation>
```

6.11.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
   <address>tel:+19585550103</address>
   <name>John</name>
   <status>Invited</status>
    <clientCorrelator>12345</clientCorrelator>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
  </resourceURL>
</chat:participantInformation>
```

6.11.5.2 Example 2: Adding multiple Participants to a group chat (Informative)

6.11.5.2.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
```

```
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantList  xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
 <participant>
   <address>tel:+19585550103</address>
   <name>John</name>
  <clientCorrelator>12345</clientCorrelator>
 </participant>
 <participant>
   <address>tel:+19585550104</address>
   <name>Peter</name>
  <clientCorrelator>67890</clientCorrelator>
 </participant>
</chat:participantList>
```

6.11.5.2.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantList  xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
 <participant>
   <address>tel:+19585550100</address>
   <name>Alice</name>
   <isOriginator>true</isOriginator>
   <status>Connected</status>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550101</address>
   <name>Bob</name>
   <status>Connected</status>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550102</address>
   <name>Ted</name>
   <status>Connected</status>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
  </resourceURL>
 </participant>
```

```
 <participant>
   <address>tel:+19585550103</address>
   <name>John</name>
   <status>Invited</status>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
  </resourceURL>
 </participant>
 <participant>
   <address>tel:+19585550104</address>
   <name>Peter</name>
  <status>Invited</status>
  <clientCorrelator>67890</clientCorrelator>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part005
  </resourceURL>
 </participant>
 <resourceURL>
   http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
  </resourceURL>
</chat:participantList>
```

6.11.5.3 Example 3: Error situation when trying to re-join a group chat session (Informative)

This example illustrates the case that the client is not allowed to re-join a group chat session, or that the session does not exist. Either error code 404 (for non-existing sessions) or 403 (for sessions to which the client has no access) are returned.

6.11.5.3.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/xml
Host: example.com
```

6.11.5.3.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="ParticipantList"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants"/>
  <policyException>
    <messageId>POL1011</messageId>
    <text>Access denied.</text>
  </policyException>
</common:requestError>
```

6.11.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC 2616].

6.12 Resource: Individual Participant in a group chat session

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}

This resource represents a Participant in a group chat session.

6.12.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session This identifier is populated by the server during resource creation.
participantId	Identifier of the Participant Note that this Id is assigned by the server upon resource creation.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.12.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.12.3 GET

This operation is used to retrieve information about an individual group chat Participant .

6.12.3.1 Example: Retrieving information about an individual group chat Participant (Informative)

6.12.3.1.1 Request

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.12.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
   <address>tel:+19585550103</address>
   <name>John</name>
   <status>Invited</status>
    <clientCorrelator>12345</clientCorrelator>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
  </resourceURL>
</chat:participantInformation>
```

6.12.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.12.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.12.6 DELETE

This operation ends the participation of a Participant in the group chat session, i.e. disconnects the Participant from the session.

It is used in the following contexts:

- by the Originator to remove a Participant from the chat session (OPTIONAL, and subject to service provider policies)
- by the Terminating Participant to decline an invitation to a chat session
- by any Participant to leave the chat session.

Note that a Participant who has left the session can re-join (if allowed by policies) using the mechanism defined in section 6.11.5.

Also note that if the Originator leaves, this may lead to the session being torn down by the server, depending on service provider policies.

As a result of performing the DELETE operation, the server SHALL remove the {participantId} node of the removed Participant from the resource tree, but SHALL keep the "sessionId" node and its "participants" sub-node available for a certain period of time that is controlled by policies. As it is not guaranteed that the server will receive information regarding the further session progress after leaving the session, GET access to these resources on behalf of a disconnected Participant SHALL return '204 No Content'.

6.12.6.1 Example: Leaving a group chat session (Informative)

6.12.6.1.1 Request

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.12.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

6.13 Resource: Individual group chat session Participant status

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/participants/{participantId}/status

This resource represents the status of a Participant in a group chat session and is used for accepting a group chat invitation, by means of updating the status.

6.13.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session

	This identifier is populated by the server during resource creation.
participantId	Identifier of the Participant Note that this Id is assigned by the server upon resource creation.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.13.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.13.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: PUT' field in the response as per section 14.7 of [RFC 2616].

6.13.4 PUT

This operation is used is used for accepting a group chat invitation, by means of updating the status. As this is a POST request that leads to an update rather than to a child resource creation, the response code on success is 200 OK.

6.13.4.1 Example 1: Accepting a group chat invitation (Informative)

6.13.4.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantSessionStatus xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
    <status>Connected</status>
</chat:participantSessionStatus>
```

6.13.4.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantSessionStatus xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <status>Connected</status>
</chat:participantSessionStatus>
```

Note that the pendant operation, i.e. declining a group chat invitation, is the same as leaving a group chat session. For an example see section 6.12.6.1

6.13.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: PUT' field in the response as per section 14.7 of [RFC 2616].

6.13.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: PUT' field in the response as per section 14.7 of [RFC 2616].

6.14 Resource: Chat messages in a group chat session

The resource used is:

http://{serverRoot}/chat/{apiVersion}/{userId}/group/{sessionId}/messages

This resource represents the set of messages in a group chat session.

In the current version of the specification, there is no difference between the chat messages resource tree in 1-1 and group chat scenarios, apart from the structure of the resource URL.

6.14.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section 5.1.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the chat session This identifier is populated by the server during resource creation.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.14.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.14.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.14.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.14.5 POST

This operation is used to create a chat message. This method MUST return either a resourceReference root element or a chatMessage root element, where using the first option is RECOMMENDED.

6.14.5.1 Example 1: Creating a group chat message, using tel URI and returning the location of the created resource (Informative)

6.14.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <text>How are you?</text>
  <reportRequest>Displayed</reportRequest>
</chat:chatMessage>
```

6.14.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
 <resourceURL>
  http://example.com/exampleAPI/chat/v1/ tel%3A%2B19585550100/group/sess001/messages/msg001
 </resourceURL>
</common:resourceReference>
```

Note that alternatively, a copy of the created resource can be returned, as illustrated in section 6.7.5.2.

6.14.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.15 Resource: Client notification containing incoming message

This resource is a callback URL provided by the client for notifications about incoming messages. The actual messages are inlined in the notifications.

The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described in section 6.15.5.

To message notifications in Confirmed and Ad-hoc 1-1 Chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/oneToOne
n/a	MessageNotification	Receiver	PUT(Displayed)	MessageStatusReport	/{otherUserId}/{sessionId}/messages/{messageId}/status

To message notifications in group chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/group
n/a	MessageNotification	Receivers	n/a	ChatMessage	/{sessionId}/messages/{messageId}

6.15.1 Request URL variables

Client provided if any.

6.15.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see see section 7.

6.15.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.15.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.15.5 POST

This operation is used to notify the client about incoming messages, and to deliver these messages to the client.

6.15.5.1 Example: Notify a client about incoming messages (Informative)

6.15.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:messageNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="ChatMessage"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001"/>
  <senderAddress>tel:+19585550102</senderAddress>
  <senderName>Ted</senderName>
  <chatMessage>
    <text>Hello Alice</text>
    <reportRequest>Displayed</reportRequest>
    <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001</resourceURL>
  </chatMessage>
  <dateTime>2001-12-17T09:30:47Z</dateTime>
</chat:messageNotification>
```

6.15.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.15.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.16 Resource: Client notification about message status

This resource is a callback URL provided by the client for notifications about message status such as “Delivered”, “Failed”, “Displayed”.

The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.16.5.

This resource is not relevant in group chats.

To message status notifications in Confirmed and Ad-hoc 1-1 Chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/oneToOne
n/a	MessageStatusNotification	Sender	n/a	ChatMessage	/{otherUserId}/{sessionId}/messages/{messageId}

6.16.1 Request URL variables

Client provided if any.

6.16.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.16.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.16.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.16.5 POST

This operation is used to notify the client the status of a message that it has sent. It is only relevant for 1-1 chats.

6.16.5.1 Example: Notify a client about message status (Informative)

6.16.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:messageStatusNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="ChatMessage"

  href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001"/>
  <status>Displayed</status>
</chat:messageStatusNotification>
```

6.16.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.16.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.17 Resource: Client notification about 1-1 chat session invitations

This resource is a callback URL provided by the client for notification about chat session invitations. The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.17.5.

This resource is not relevant in Ad-hoc 1-1 Chats and group chats.

To chat session invitation notifications in Confirmed 1-1 Chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/oneToOne
n/a	SessionInvitationNotification	Terminating Participant	PUT(accept) DELETE	ChatSessionInformation	/{otherUserId}/{sessionId}

6.17.1 Request URL variables

Client provided if any.

6.17.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.17.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.17.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.17.5 POST

This operation is used to notify the client about chat session invitations.

6.17.5.1 Example: Notify a client about 1-1 chat session invitations (Informative)

6.17.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:sessionInvitationNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="ChatSessionInformation"
     href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550101/oneToOne/tel%3A%2B19585550100/sess001"/>
   <subject>Hi there!</subject>
   <originatorAddress>tel:+19585550100</originatorAddress>
   <originatorName>Alice</originatorName>
   <tParticipantAddress>tel:+19585550101</tParticipantAddress>
   <tParticipantName>Bob</tParticipantName>
</chat:sessionInvitationNotification>
```

6.17.5.1.2 Response

HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT

6.17.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.18 Resource: Client notification about group chat session invitations

This resource is a callback URL provided by the client for notification about chat session invitations. The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.18.5.

This resource is not relevant in 1-1 chats.

To chat session invitation notifications in group chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/group
n/a	GroupSessionInvitation Notification	Terminating Participants	PUT(accept) DELETE	GroupChatSessionInformation ParticipantInformation	/{sessionId} /{sessionId}/participants/{participantId}/

6.18.1 Request URL variables

Client provided if any.

6.18.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.18.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.18.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.18.5 POST

This operation is used to notify the client about chat session invitations.

6.18.5.1 Example: Notify a client about group chat session invitations(Informative)

This example notification is triggered by the request in example 6.9.5.1. Note that the {userId} resourceURL variable represents the userId of the user on whose behalf the application acts, not the one of the Originator.

6.18.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:groupSessionInvitationNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="GroupChatSessionInformation"
     href=" http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550102/group/sess001"/>
  <link rel=" ParticipantInformation"
     href=" http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550102/group/sess001/participants/part003"/>
  <subject>Hello</subject>
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <isOriginator>true</isOriginator>
    <status>Connected</status>
    <resourceURL>
       http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
     </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
    <status>Invited</status>
    <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
     </resourceURL>
```

```
  </participant>
  <participant>
    <address>tel:+19585550102</address>
    <name>Ted</name>
    <status>Invited</status>
    <resourceURL>
       http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
    </resourceURL>
  </participant>
</chat:groupSessionInvitationNotification>
```

6.18.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.18.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.19 Resource: Client notification about chat session events

This resource is a callback URL provided by the client for notification about various chat session events. The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.19.5.

This resource is not relevant in Ad-hoc 1-1 Chats.

To chat session event notifications in Confirmed 1-1 Chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/oneToOne
Accepted	ChatEventNotification	Originator	n/a	ChatSessionInformation	/{otherUserId}/{sessionId}
Declined	ChatEventNotification	Originator	n/a	ChatSessionInformation	/{otherUserId}/{sessionId}

SessionCancelled	ChatEventNotification	Participants	n/a	ChatSessionInformation	/{otherUserId}/{sessionId}
SessionEnded	ChatEventNotification	Participants	n/a	ChatSessionInformation	/{otherUserId}/{sessionId}

To chat session event notifications in group chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}/group
SessionEnded	ChatEventNotification	Participants	n/a	GroupChatSessionInformation	/{sessionId}

6.19.1 Request URL variables

Client provided if any.

6.19.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.19.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.19.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.19.5 POST

This operation is used to notify the client about chat session invitations.

6.19.5.1 Example: Notify a client about chat session events (Informative)

6.19.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatEventNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="GroupChatSessionInformation"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001"/>
  <link rel="ChatNotificationSubscription"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
  <eventType>SessionEnded</eventType>
  <eventDescription>The session has ended.</eventDescription>
</chat:chatEventNotification>
```

6.19.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.19.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.20 Resource: Client notification about changes of Participant status

This resource is a callback URL provided by the client for notification about changes of Participant status. The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.20.5.

The notification is sent by the server to all subscribed Participants in the chat session triggered by Participants re-joining or leaving the chat.

This resource is not relevant in 1-1 chats.

To participant status notifications in group chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL:

					//{serverRoot}/chat/{apiVersion}/{userId}/group
n/a	ParticipantStatusNotification	Participants	n/a	GroupChatSessionInformation	/{sessionId}

6.20.1 Request URL variables

Client provided if any.

6.20.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.20.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.20.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.20.5 POST

This operation is used to notify the client about changes of Participant status.

6.20.5.1 Example: Notify a client about Participant status changes (Informative)

6.20.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantStatusNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
  <link rel="GroupChatSessionInformation"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001"/>
  <link rel="ChatNotificationSubscription"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
```

```
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <status>Connected</status>
    <yourown>true</yourown>
    <link rel="ParticipantInformation"
      href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001"/>
  </participant>
  <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
    <status>Disconnected</status>
    <yourown>false</yourown>
    <link rel="ParticipantInformation"
      href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002"/>
  </participant>
</chat:participantStatusNotification>
```

6.20.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.20.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.21 Resource: Client notification about subscription cancellation

This resource is a callback URL provided by the client for notification about subscription cancellations, which are usually due to the subscription expiring. The RESTful Chat API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.20.5.

The notification is sent by the server to the Participant to whom the cancelled subscription belongs.

To subscription cancellation notifications in group chats, the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/chat/{apiVersion}/{userId}
n/a	SubscriptionCancellationNotification	subscriber	n/a	ChatNotificationSubscription	/subscriptions/{subscriptionId}

6.21.1 Request URL variables

Client provided if any.

6.21.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful Chat API, see section 7.

6.21.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.21.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.21.5 POST

This operation is used to notify the client about changes of Participant status.

6.21.5.1 Example: Notify a client about subscription cancellation (Informative)

6.21.5.1.1 Request

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<chat:subscriptionCancellationNotification xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackData>abcd</callbackData>
```

```
  <link rel="ChatNotificationSubscription"
    href="http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
</chat:subscriptionCancellationNotification >
```

6.21.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.21.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common].

There are no additional specific Service Exception codes defined for this release of the Chat API.

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common].

The following additional Policy Exception codes are defined for the Chat API.

7.2.1 POL1012: Messages during session setup not supported

Name	Description
MessageID	POL1012
Text	Messages during session setup not supported.
Variables	None
HTTP status code(s)	403 Forbidden

7.2.2 POL1013: Confirmed 1-1 chats not supported

Name	Description
MessageID	POL1013
Text	Confirmed 1-1 chats are not supported.
Variables	None
HTTP status code(s)	403 Forbidden

7.2.3 POL1014: Ad-hoc 1-1 chats not supported

Name	Description
MessageID	POL1014
Text	Ad-hoc 1-1 chats are not supported.
Variables	none
HTTP status code(s)	403 Forbidden

7.2.4 POL1017: Too many participants

Name	Description
MessageID	POL1017
Text	Too many participants.
Variables	none
HTTP status code(s)	403 Forbidden

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Version OMA-TS-REST_NetAPI _Chat-V1_0	28 Apr 2011	All	TS skeleton created
	31 May 2011	Many	OMA-ARC-REST-NetAPI-2011-0020R05-CR_Chat_API_basic_design implemented.
	07 Jul 2011	Many	OMA-ARC-REST-NetAPI-2011-0124R03-CR_Chat_API_resource_and_datatype_alignment_with_new_resource_model implemented.
	27 Jul 2011	2.1 5.1 5.2.2.x 5.2.3.x	Implemented CRs - OMA-ARC-REST-NetAPI-2011-0156-CR_Chat_alignment_with_FT_IS_VS - OMA-ARC-REST-NetAPI-2011-0157R01-CR_Chat_small_fix
	02 Aug 2011	Many	OMA-ARC-REST-NetAPI-2011-0173R01-CR_Chat_section_6_structure_with_tel_URI_fixes_and_Notif_channel_changes implemented
	08 Sep 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0197R01-CR_Chat_Appendix_C - OMA-ARC-REST-NetAPI-2011-0093R03-CR_Chat_Flows - OMA-ARC-REST-NetAPI-2011-0220-CR_Separating_originator_and_tParticipant_1_1_chat^ - OMA-ARC-REST-NetAPI-2011-0227R02-CR_Chat_Long_Polling_fix
	26 Sep 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0238R02-CR_ACR_Chat - OMA-ARC-REST-NetAPI-2011-0250-CR_ChatEventNotification_fix
	17 Oct 2011	5.2.3.3, 5.2.2.11, 5.2.2.12	CR implemented: OMA-ARC-REST-NetAPI-2011-0275R01-CR_Chat_status_enum_fix
	11 Nov 2011	Many	CRs implemented: - OMA-ARC-REST-NetAPI-2011-0284R02-CR_Simplifying_1_1_chat_sessions - OMA-ARC-REST-NetAPI-2011-0237R02-CR_Chat_examples Note that CR 284 implements a fundamental change in the approach to 1-1 chats. Rather than exposing session-based 1-1 chats only, this now supports both session-based and session-less 1-1 chats. The new mode greatly simplifies the API, and makes it closer to today's chat APIs in the Internet. Session-based oneToOne chats are supposed to be optional.
	18 Nov 2011	Many	CR OMA-ARC-REST-NetAPI-2011-0375-CR_Chat_actions_and_editorials implemented

Document Identifier	Date	Sections	Description
	23 Nov 2011	Many	CRs implemented: - OMA-ARC-REST-NetAPI-2011-0403R01-CR_Chat_JSON - OMA-ARC-REST-NetAPI-2011-0409-CR_Annex_F_in_Chat - OMA-ARC-REST-NetAPI-2011-0410-CR_Section_4_in_Chat - OMA-ARC-REST-NetAPI-2011-0412R01-CR_Chat_Flows_updates Editorial: - Converted Word comments into editor's notes - Removed pure-editorial editor's notes
	29 Nov 2011	Many	CRs implemented - OMA-ARC-REST-NetAPI-2011-0411R01-CR_Chat_SCR_tables - OMA-ARC-REST-NetAPI-2011-0416-CR_Chat_Appendix_G - OMA-ARC-REST-NetAPI-2011-0424-INP_HTML_401_reference_blueprint Editorial: - Removed hyperlinks in JSON examples
	22 Dec 2011	None	Repackaged as the previous revision was stored in a broken ZIP file. No changes to actual content.
	26 Jan 2012	Many	Implemented CRs: - OMA-ARC-REST-NetAPI-2012-0013-CR_TS_Chat_CONR_editorials_and_simple_bugfixes - OMA-ARC-REST-NetAPI-2011-0434R01-INP_Blueprint_for_APIs_changes_for_ACR_Authorization - OMA-ARC-REST-NetAPI-2012-0029-INP_NOTIFY_blueprint
	22 Feb 2012	G.1.2	Editorial: capitalized "Authorization" in "acr:Authorization"
	13 Mar 2012	Many	Implemented OMA-ARC-REST-NetAPI-2012-0102R01-CR_Chat_TS_Addressing_new_global_comments
	27 Mar 2012	Many	CRs implemented: - OMA-ARC-REST-NetAPI-2012-0106R01-CR_Chat_implementing_new_resource_structure - OMA-ARC-REST-NetAPI-2012-0114-CR_Chat_Notifications_with_TTL_blueprint
	05 Apr 2012	Some	Implemented OMA-ARC-REST-NetAPI-2012-0126R01-CR_Session_aware_session_unaware_signalling_Chat
	16 Apr 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0146R02-CR_Chat_section_5_3_intro - OMA-ARC-REST-NetAPI-2012-0145R01-CR_Chat_even_more_CONR_resolutions - OMA-ARC-REST-NetAPI-2012-0142R01-CR_Chat_more_CONR_resolutions - OMA-ARC-REST-NetAPI-2012-0131R01-CR_Chat_flows_session_aware
	17 Apr 2012	Many	OMA-ARC-REST-NetAPI-2012-0149R01-CR_Chat_loose_ends implemented
	18 Apr 2012	Many	OMA-ARC-REST-NetAPI-2012-0152R01-CR_Chat_session_aware_session_unaware_take_3 implemented
	24 Apr 2012	Many	CRs implemented - OMA-ARC-REST-NetAPI-2012-0159-CR_Chat_errorcodes - OMA-ARC-REST-NetAPI-2012-0158R01-CR_Chat_cleanup

Document Identifier	Date	Sections	Description
	02 May 2012	Many	CR implemented - OMA-ARC-REST-NetAPI-2012-0163-CR_Chat_TS_address_example_validation_errors
	03 May 2012	All	Suffix "-v1" removed from document name Editorial changes
Candidate Version OMA-TS-REST_NetAPI _Chat-V1_0	09 May 2012	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2012-0195-INP_REST_NetAPI_Chat_1_0_ERP_and_ETR_for_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for REST.Chat Server

Item	Function	Reference	Requirement
REST-CHAT-SUPPORT-S-001-M	Support for the RESTful Chat API	5, 6	
REST-CHAT-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-CHAT-SUPPORT-S-003-M	Support for the JSON request & response format	6	
REST-CHAT-SUPPORT-S-004-O	Support for the application/x-www-form-urlencoded format	Appendix C	

B.1.1 SCR for REST.CHAT.Subscriptions Server

Item	Function	Reference	Requirement
REST-CHAT-SUBSCR-S-001-M	Support for subscriptions to chat event notifications	6.1	
REST-CHAT-SUBSCR-S-002-O	Read the list of active subscriptions to chat event notifications – GET	6.1.3	
REST-CHAT-SUBSCR-S-003-M	Create new subscription to chat event notifications – POST (XML or JSON)	6.1.5	
REST-CHAT-SUBSCR-S-004-O	Create new subscription to chat event notifications – POST (application/x-www-form-urlencoded)	C.1	

B.1.2 SCR for REST.CHAT.IndSubscription Server

Item	Function	Reference	Requirement
REST-CHAT-INDSUBSCR-S-001-M	Support for access to an individual subscription to chat event notifications	6.2	
REST-CHAT-INDSUBSCR-S-002-O	Read an individual subscription to chat event notifications – GET	6.2.3	

Item	Function	Reference	Requirement
REST-CHAT-INDSUBSCR-S-003-M	Cancel subscription and stop corresponding notifications – DELETE	6.2.6	

B.1.3 SCR for REST.CHAT.OneToOne.Sessions Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-SESS-S-001-M	Support for 1-1 chat sessions between two users	6.3	REST-CHAT-ONE2ONE-INDSESS-CONF-S-001-O OR REST-CHAT-ONE2ONE-INDSESS-ADH-S-001-O
REST-CHAT-ONE2ONE-SESS-S-002-O	Creating a 1-1 chat session between two users – POST (XML or JSON)	6.3.5	REST-CHAT-ONE2ONE-INDSESS-CONF-S-001-O
REST-CHAT-ONE2ONE-SESS-S-003-O	Creating a 1-1 chat session between two users – POST (application/x-www-form-urlencoded)	C.4	REST-CHAT-ONE2ONE-INDSESS-CONF-S-001-O

B.1.4 SCR for REST.CHAT.OneToOne.IndSession.Confirmed Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-INDSESS-CONF-S-001-O	Support for individual Confirmed 1-1 Chats	6.4	REST-CHAT-ONE2ONE-INDSESS-CONF-S-003-O
REST-CHAT-ONE2ONE-INDSESS-CONF-S-002-O	Read individual 1-1 chat session information – GET	6.4.3	
REST-CHAT-ONE2ONE-INDSESS-CONF-S-003-O	Cancel invitation / Decline Invitation / Terminate an individual 1-1 chat session – DELETE	6.4.6	

B.1.5 SCR for REST.CHAT.OneToOne.IndSession.Adhoc Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-INDSESS-ADH-S-001-O	Support for Ad-hoc 1-1 Chats	6.4	
REST-CHAT-ONE2ONE-INDSESS-ADH-S-002-O	Read individual 1-1 chat session information – GET	6.4.3	

B.1.6 SCR for REST.CHAT.OneToOne.IndSession.Status Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-INDSESS-STAT-S-001-O	Support for 1-1 chat session status	6.5	REST-CHAT-ONE2ONE-INDSESS-STAT-S-002-O
REST-CHAT-ONE2ONE-INDSESS-STAT-S-002-O	Accept a 1-1 chat session invitation – POST(XML or JSON)	6.5.4	

B.1.7 SCR for REST.CHAT.OneToOne.IndSession.Extend Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-INDSESS-EXT-S-001-O	Support for extending a 1-1 chat to a group chat session	6.6	REST-CHAT-ONE2ONE-INDSESS-EXT-S-002-O
REST-CHAT-ONE2ONE-INDSESS-EXT-S-002-O	Extend a 1-1 chat to a group chat session – POST (XML or JSON)	6.6.5	
REST-CHAT-ONE2ONE-INDSESS-EXT-S-003-O	Extend a 1-1 chat to a group chat session – POST(application/x-www-form-urlencoded)	C.4	

B.1.8 SCR for REST.CHAT.OneToOne.Messages Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-MSG-S-001-M	Support for chat messages in a 1-1 chat	6.7	
REST-CHAT-ONE2ONE-MSG-S-002-M	Create (send) a 1-1 chat message – POST (XML or JSON)	6.7.5	
REST-CHAT-ONE2ONE-MSG-S-003-O	Create (send) a 1-1 chat message – POST (application/x-www-form-urlencoded)	C.2	

B.1.9 SCR for REST.CHAT.OneToOne.IndMessage.Status Server

Item	Function	Reference	Requirement
REST-CHAT-ONE2ONE-INDMSG-STAT-S-001-M	Support for individual message status in a 1-1 chat	6.8	
REST-CHAT-ONE2ONE-INDMSG.STAT-S-002-O	Read the status of a chat message – GET (XML or JSON)	6.8.3	
REST-CHAT-ONE2ONE-	Report the status of a chat message – PUT	6.8.4	

Item	Function	Reference	Requirement
INDMSG.STAT-S-003-M	(XML or JSON)		

B.1.10 SCR for REST.CHAT.Group.Sessions Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-SESS-S-001-M	Support for group chat sessions	6.9	
REST-CHAT-GROUP-SESS-S-002-M	Create a new group chat session – POST(XML or JSON)	6.9.5	
REST-CHAT-GROUP-SESS-S-003-O	Create a new group chat session – POST(application/x-www-form-urlencoded)	C.6	

B.1.11 SCR for REST.CHAT.Group.IndSession Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-INDSESS-S-001-O	Support for individual group chat sessions	6.10	
REST-CHAT-GROUP-INDSESS-S-002-O	Retrieve information about an individual group chat session – GET	6.10.3	
REST-CHAT-GROUP-INDSESS-S-003-O	Terminate individual group chat session – DELETE	6.10.6	

B.1.12 SCR for REST.CHAT.Group.IndSession.Participants Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-INDSESS-PART-S-001-M	Support for Participants in a group chat session	6.11	
REST-CHAT-GROUP-INDSESS-PART-S-002-O	Read the list of all Participants in a group chat session – GET	6.11.3	
REST-CHAT-GROUP-INDSESS-PART-S-003-M	Add one or more Participants to a group chat session (invite, re-join) – POST (XML or JSON)	6.11.5	
REST-CHAT-GROUP-INDSESS-PART-S-004-O	Add one or more Participants to a group chat session (invite, re-join) – POST	C.7	

Item	Function	Reference	Requirement
	(application/x-www-form-urlencoded)		

B.1.13 SCR for REST.CHAT.Group.IndSession.IndParticipant Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-INDSESS-INDPART-S-001-M	Support for individual Participants in a group chat session	6.12	
REST-CHAT-GROUP-INDSESS-INDPART-S-002-O	Read information about an individual Participant in a group chat session – GET	6.12.3	
REST-CHAT-GROUP-INDSESS-INDPART-S-003-M	Remove an individual Participant from a group chat session (leave); or decline invitation – DELETE	6.12.6	

B.1.14 SCR for REST.CHAT.Group.IndSession.IndParticipant. Status Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-INDSESS-INDPART-S-001-M	Support for the status of an individual group chat session Participant	6.13	
REST-CHAT-GROUP-INDSESS-INDPART-S-003-M	Accept group chat session invitation – PUT (XML or JSON)	6.13.4	

B.1.15 SCR for REST.CHAT.Group.Messages Server

Item	Function	Reference	Requirement
REST-CHAT-GROUP-MSG-S-001-M	Support for chat messages in a group chat	6.14	
REST-CHAT-GROUP-MSG-S-002-M	Create (send) a group chat message – POST (XML or JSON)	6.14.5	
REST-CHAT-GROUP-MSG-S-003-O	Create (send) a group chat message – POST (application/x-www-form-urlencoded)	C.2	

B.1.16 SCR for REST.CHAT.Notifications.Message Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF	Support for	6.15	

Item	Function	Reference	Requirement
MSG-S-001-M	notifications containing incoming chat message		
REST-CHAT-NOTIF-MSG-S-002-M	Notification containing incoming chat message – POST (XML or JSON)	6.15.5	

B.1.17 SCR for REST.CHAT.Notifications.Message.Status Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-MSG-STAT-S-001-M	Support for notifications about message status	6.16	
REST-CHAT-NOTIF-MSG-STAT-S-002-M	Notification about message status – POST (XML or JSON)	6.16.5	

B.1.18 SCR for REST.CHAT.Notifications.OneToOne.Invite Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-ONE2ONE-INVITE-S-001-O	Support for notifications about 1-1 chat session invitations	6.15	REST-CHAT-NOTIF-ONE2ONE-INVITE-S-002-O
REST-CHAT-NOTIF-ONE2ONE-INVITE-S-002-O	Notification about 1-1 chat session invitation – POST (XML or JSON)	6.17.5	

B.1.19 SCR for REST.CHAT.Notifications.Group.Invite Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-GROUP-INVITE-S-001-M	Support for notifications about group chat session invitations	6.18	
REST-CHAT-NOTIF-GROUP-INVITE-S-002-M	Notification about group chat session invitation – POST (XML or JSON)	6.18.5	

B.1.20 SCR for REST.CHAT.Notifications.Events Server

Item	Function	Reference	Requirement

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-EVENT-S-001-M	Support for notifications about chat session events	6.19	
REST-CHAT-NOTIF-EVENT-S-002-M	Notification about chat session event – POST (XML or JSON)	6.19.5	

B.1.21 SCR for REST.CHAT.Notifications.Group.Participants Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-GROUP-PART-S-001-M	Support for notifications about changes of Participant status	6.20	
REST-CHAT-NOTIF-GROUP-PART-S-002-M	Notification about changes of Participant status – POST (XML or JSON)	6.20.5	

B.1.22 SCR for REST.CHAT.Notifications.SubscriptionCancellation Server

Item	Function	Reference	Requirement
REST-CHAT-NOTIF-SUBCXL-S-001-M	Support for notifications about subscription cancellation	6.21	
REST-CHAT-NOTIF-SUBCXL-S-002-M	Notification about subscription cancellation – POST (XML or JSON)	6.21.5	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This section defines a format for the RESTful Chat API requests where the body of the request is encoded using the application/x-www-form-urlencoded MIME type.

Note: only the request body is encoded as application/x-www-form-urlencoded, the response is still encoded as XML or JSON depending on the preference of the client and the capabilities of the server. Names and values MUST follow the application/x-www-form-urlencoded character escaping rules from [W3C_URLENC].

The encoding is defined below for all REST operations in the Chat API which are based on POST requests, except Notifications.

C.1 Creating a new subscription to chat notifications

This operation is used to create a new subscription to chat notifications. See section 6.1.5.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/Values	Optional	Description
notifyURL	xsd:anyURI	No	Notification endpoint definition
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications
notificationFormat	common:NotificationFormat	Yes	Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}. Default: XML
confirmedChatSupported	xsd:boolean	Yes	In resource-creating requests, this flag signals to the server that this client supports Confirmed 1-1 Chats. In case this is present and set to true, the client supports Confirmed 1-1 Chats. In the created resource, the server sets this flag to true in case it was set to true by the client in the corresponding creation request and the server supports Confirmed 1-1 Chats; otherwise the server either sets it to false or omits it. If the server does not support any of the

			methods signalled by the client using the elements "confirmedChatSupported" and "adhocChatSupported", it SHALL reject the subscription either with the exception POL1013 (if Confirmed 1-1 Chat are not supported) or with the exception POL1014 (if Ad-hoc 1-1 Chats are not supported). Default: false
adhocChatSupported	xsd:boolean	Yes	In resource-creating requests, this flag signals to the server that this client supports Ad-hoc 1-1 Chats. In case this is absent or set to true, the client supports for Ad-hoc 1-1 Chats. In the created resource, the server sets this flag to true or omits it in case it was absent or set to true in the corresponding creation request and the server supports Ad-hoc 1-1 Chats; otherwise the server sets it to false. If the server does not support any of the methods signalled by the client using the elements "confirmedChatSupported" and "adhocChatSupported", it SHALL reject the subscription either with the exception POL1013 (if Confirmed 1-1 Chats are not supported) or with the exception POL1014 (if Ad-hoc 1-1 Chats are not supported). Default: true Note: the default is "true" here for maximum simplification of the API.
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to "0" (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the desired lifetime of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.

clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscriptions in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.1.1 Example (Informative)

C.1.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

notifyURL=http%3A%2F%2Fapplication.example.com%2Fchat%2Fnotifications%2F77777&
callbackData=abcd&
duration=7200&
clientCorrelator=12345
```

C.1.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatNotificationSubscription xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
  <callbackReference>
    <notifyURL>http://application.example.com/chat/notifications/77777</notifyURL>
    <callbackData>abcd</callbackData>
  </callbackReference>
```

```
  <duration>7200</duration>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</chat:chatNotificationSubscription>
```

C.2 Creating a chat message

This operation is used to create a chat message in a 1-1 chat, see section 6.7.5, or in a group chat session, see section 6.14.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
text	xsd:string	No	Text content of a chat message
reportRequest	MessageStatus[0..unbounded]	Yes	Request message status reports (i.e. the Receiver of the message should report message delivery / disposition status) Note that the underlying system might not support reporting.

If the operation was successful, it returns an HTTP Status of "201 Created", and MUST return either a resourceReference root element or a chatMessage root element, where using the first option is RECOMMENDED.

C.2.1 Example 1: using tel URI and returning the location of the created resource (Informative)

Note that the example assumes a 1-1 chat. Posting a chat message to a group chat session looks the same, apart from using a different resource URL.

Further note that alternatively, a copy of the created resource can be returned.

C.2.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

text=How%20are%20you%3F&
reportRequest=Displayed
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages/msg001
```

```
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common ="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages/msg001
  </resourceURL>
</common:resourceReference>
```

C.2.2 Example 2: using ACR and returning a copy of the created resource (Informative)

Note that the example assumes a 1-1 chat. Posting a chat message to a group chat session looks the same, apart from using a different resource URL.

C.2.2.1 Request

```
POST /exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/messages
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

text=How%20are%20you%3F&
reportRequest=Displayed
```

C.2.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/messages/msg001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatMessage  xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <text>How are you?</text>
  <reportRequest>Displayed</reportRequest>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/messages/msg001
   </resourceURL>
</chat:chatMessage >
```

C.3 Creating an “isComposing” message

This operation is used to send an “isComposing” message, see section 6.7.5.3

The request parameters are as follows:

Name	Type/Values	Optional	Description
state	xsd:string	No	Sender state, as defined in [RFC3994]. One of “idle”, “active”
lastactive	xsd:dateTime	Yes	Time of last activity, as defined in [RFC3994]
contenttype	xsd:string	Yes	Type of message being created, as defined in [RFC3994] This element contains either a MIME media type, or a combination of media type and subtype.
refresh	xsd:positiveInteger	Yes	Time interval in seconds after which the Receiver can expect an update from the Sender, as defined in [RFC3994]
(any)	any[0..unbounded]	Yes	Any element from another namespace, as defined in [RFC3994]

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.3.1 Example (Informative)

C.3.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

state=active&
contenttype=text%2Fplain&
refresh=90
```

C.3.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages/msg002
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/ tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/messages/msg002
  </resourceURL>
```

```
</common:resourceReference>
```

C.4 Creating a 1-1 chat session

This operation is used to create a 1-1 chat session with the user represented by {userId} as Originator and the one represented by {otherUserId} as Terminating Participant. See section 6.3.5.

For Ad-hoc 1-1 Chats (i.e. those using the reserved value “adhoc” for {sessionId}), this step is not necessary.

The request parameters are as follows:

Name	Type/Values	Optional	Description
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
originatorAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator If originatorAddress is also part of the request URL, the two MUST have the same value.
originatorName	xsd:string	Yes	Human readable name of the Originator
tParticipantAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Terminating Participant If tParticipantAddress is also part of the request URL, the two MUST have the same value.
tParticipantName	xsd:string	Yes	Human readable name of the Terminating Participant

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.4.1 Example: Creating a 1-1 chat session (Informative)

C.4.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101 HTTP/1.1
Accept: application/xml
Content-Type: application/x-www-formurlencoded
Host: example.com
Content-Length: nnnn

subject=Hi there!&
originatorAddress=tel%3A%2B19585550100&
originatorName=Alice&
tParticipantAddress=tel%3A%2D19585550101&
tParticipantName=Bob
```

C.4.1.2 Response

```
HTTP/1.1 201 Created
```

```
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001

<?xml version="1.0" encoding="UTF-8"?>
<chat:chatSessionInformation xmlns:chat="urn:oma:xml:rest:netapi:chat:1">
   <subject>Hi there!</subject>
   <originatorAddress>tel:+19585550100</originatorAddress>
   <originatorName>Alice</originatorName>
   <tParticipantAddress>tel:+19585550101</tParticipantAddress>
   <tParticipantName>Bob</tParticipantName>
   <status>Invited</status>
   <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001
   </resourceURL>
</chat:chatSessionInformation>
```

C.5 Extending a 1-1 chat to a group chat session

This operation is used to extend a one-to-one chat session to a group chat session. See section 6.6.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
tParticipantAddress	xsd:anyURI [1..unbounded]	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Terminating Participants (i.e. invited users) to be added to the session. If tParticipantAddress is also part of the request URL, the two MUST have the same value.
tParticipantName	xsd:string [0..unbounded]	Yes	Human readable name of the Terminating Participants (i.e. invited users)
clientCorrelator	xsd:string[0..unbounded]	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids the duplicate addition of the same user to a chat session in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of “303 See Other”.

C.5.1 Example (Informative)

C.5.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550100/sess001/extend
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

tParticipantAddress= tel%3A%2B19585550102&
tParticipantName=Ted&
clientCorrelator=ABCDE
```

C.5.1.2 Response

```
HTTP/1.1 303 See Other
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jun 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</common:resourceReference>
```

C.6 Creating a new group chat session

This operation is used to create a new group chat session. See section 6.9.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
subject	xsd:string	No	Initial message of the chat session, passed from the Originator to the invited Participants
originatorAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator If originatorAddress is also part of the request URL, the two MUST have the same value.
originatorName	xsd:string	Yes	Human readable name of the Originator
tParticipantAddress	xsd:anyURI [1..unbounded]	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Terminating Participant (i.e. invited user)
tParticipantName	xsd:string [0..unbounded]	Yes	Human readable name of the Terminating Participant (i.e. invited user)
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this

			particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate chat session creations in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.6.1 Example (Informative)

C.6.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

subject=Hello%20friends&
originatorAddress=tel%3A%2B19585550100&
originatorName=Alice&
tParticipantAddress=tel%3A%2B19585550101&
tParticipantName=Bob&
tParticipantAddress=tel%3A%2B19585550102&
tParticipantName=Ted&
clientCorrelator=12345
```

C.6.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:groupChatSessionInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <subject>Hello friends</subject>
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <isOriginator>true</isOriginator>
    <status>Connected</status>
```

```
   <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
   </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
    <status>Invited</status>
    <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
    </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550102</address>
    <name>Ted</name>
    <status>Invited</status>
    <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
    </resourceURL>
  </participant>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001</resourceURL>
</chat:groupChatSessionInformation >
```

C.7 Adding Participant(s) to a group chat session, or re-joining a group chat session

This operation is used to add one or more Participants to a group chat session. See section 6.11.5.

It is also used to re-join a group chat session, in which case the number of additional Participants is limited to one.

The request parameters are as follows:

Name	Type/Values	Optional	Description
tParticipantAddress	xsd:anyURI [1..unbounded]	No	The address (e.g. SIP URI) of the Terminating Participant (i.e. invited user) If tParticipantAddress is also part of the request URL, the two MUST have the same value.
tParticipantName	xsd:string [0..unbounded]	Yes	Human readable name of the Terminating Participant (i.e. invited user) Number of entries MUST either be zero, or equal to the number of entries in tParticipantAddress.
clientCorrelator	xsd:string[0..unbounded]	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present.

			Note: this allows the client to recover from communication failures during resource creation and therefore avoids creating a resource twice for the same Participant in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of "201 Created" in case there was just one tParticipantAddress instance passed, or "200 OK" if there were multiple instances passed. Further, in case there was just one tParticipantAddress instance passed, the response entity body contains either a "resourceReference" or a "participant" root element. In case there were multiple tParticipantAddress instances passed, the response entity body contains a "participantList" root element.

In other words, adding one Participant corresponds to the creation of a new "participant" resource in the list of Participants and is consistent with the resource creation design pattern used throughout the OMA RESTful Network APIs, whereas the addition of multiple Participants corresponds to an update operation of the list of Participants.

C.7.1 Example 1: Adding one Participant to a group chat, or re-joining a group chat (Informative)

C.7.1.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

tParticipantAddress=tel%3A%2B19585550103&
tParticipantName=John&
clientCorrelator=123456
```

C.7.1.2 Response

```
 HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantInformation xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
   <address>tel:+19585550103</address>
   <name>John</name>
   <status>Invited</status>
    <clientCorrelator>123456</clientCorrelator>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
```

```
  </resourceURL>
</chat:participantInformation>
```

C.7.2 Example 2: Adding multiple Participants to a group chat (Informative)

C.7.2.1 Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

tParticipantAddress= tel%3A%2B19585550103&
tParticipantAddress= tel%3A%2B19585550104&
tParticipantName=John&
tParticipantName=Peter&
clientCorrelator=123456&
clientCorrelator=67890
```

C.7.2.2 Response

```
 HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<chat:participantList  xmlns:chat ="urn:oma:xml:rest:netapi:chat:1">
  <participant>
    <address>tel:+19585550100</address>
    <name>Alice</name>
    <isOriginator>true</isOriginator>
    <status>Connected</status>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001
  </resourceURL>
 </participant>
 <participant>
    <address>tel:+19585550101</address>
    <name>Bob</name>
    <status>Connected</status>
   <resourceURL>
     http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002
   </resourceURL>
 </participant>
 <participant>
    <address>tel:+19585550102</address>
    <name>Ted</name>
    <status>Connected</status>
```

```
    <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003
    </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550103</address>
    <name>John</name>
    <status>Invited</status>
    <clientCorrelator>123456</clientCorrelator>
    <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004
    </resourceURL>
  </participant>
  <participant>
    <address>tel:+19585550104</address>
    <name>Peter</name>
    <status>Invited</status>
    <clientCorrelator>67890</clientCorrelator>
    <resourceURL>
      http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part005
    </resourceURL>
  </participant>
  <resourceURL>
    http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
  </resourceURL>
</chat:participantList>
```

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a lightweight, text-based, language-independent data interchange format. It provides a simple means to represent basic name-value pairs, arrays and objects. JSON is relatively trivial to parse and evaluate using standard JavaScript libraries, and hence is suited for REST invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC 4627].

The following examples show the request and response for various operations using the JSON data format. The examples follow the XML to JSON serialization rules in [REST_NetAPI_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [REST_NetAPI_Common].

For full details on the operations themselves please refer to the section number indicated.

D.1 Reading all active chat notification subscriptions (section 6.1.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"chatSubscriptionList": {
   "chatNotificationSubscription": {
      "callbackReference": {
         "callbackData": "abcd",
         "notifyURL": "http://application.example.com/chat/notifications/77777"
      },
      "clientCorrelator": "12345",
      "duration": "7037",
      "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"
   },
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions"
}}
```

D.2 Creating a new subscription to chat notifications, response with copy of created resource (section 6.1.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
```

```
Accept: application/json
Host: example.com

{"chatNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/chat/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"chatNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/chat/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.3 Creating a new subscription to chat notifications, response with location of created resource (section 6.1.5.2)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/chat/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"resourceReference": {
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.4 Creating a new subscription to chat notifications, requiring support of Confirmed 1-1 Chats which the server does not provide (section 6.1.5.3)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatNotificationSubscription": {
   "adhocChatSupported": "false",
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/chat/notifications/77777"
   },
   "clientCorrelator": "12345",
   "confirmedChatSupported": "true",
   "duration": "7200"
}}
```

Response:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"requestError": {"policyException": {
   "messageId": "POL1013",
   "text": "Confirmed 1-1 chats are not supported."
}}}
```

D.5 Reading an individual subscription (section 6.2.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=XML HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"chatNotificationSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/chat/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration": "7200",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.6 Cancelling a subscription (section 6.2.6.1)

Request:

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

D.7 Creating a 1-1 chat session (section 6.3.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"chatSessionInformation": {
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "subject": "Hi there!",
   "tParticipantAddress": "tel:+19585550101",
   "tParticipantName": "Bob"
```

```
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001

{"chatSessionInformation": {
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001",
   "status": "Invited",
   "subject": "Hi there!",
   "tParticipantAddress": "tel:+19585550101",
   "tParticipantName": "Bob"
}}
```

D.8 Retrieving chat session information of a 1-1 session (section 6.4.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"chatSessionInformation": {
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001",
   "status": "Invited",
   "subject": "Hi there!",
   "tParticipantAddress": "tel:+19585550101",
   "tParticipantName": "Bob"
}}
```

D.9 Retrieving chat session information of a 1-1 session that was previously extended to a group chat session (section 6.4.3.2)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Response:

```
HTTP/1.1 303 See Other
Content-Type: application/json
Location: /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"resourceReference": {
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001"
}}
```

D.10 Terminating a 1-1 chat session, or declining an invitation (section 6.4.6.1)

Request:

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.11 Accepting a 1-1 chat invitation (section 6.5.4.1)

Request:

```
PUT /exampleAPI/chat/v1/tel%3A%2B19585550101/oneToOne/tel%3A%2B19585550101/sess001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"participantSessionStatus": {
   "status": "Connected"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2011 17:51:59 GMT
```

D.12 Extending a Confirmed 1-1 Chat to a group chat session (section 6.6.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/extend HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"participantList": {"participant": {
   "address": "tel:+B19585550102",
   "clientCorrelator": "ABCDE",
   "name": "Ted"
}}}
```

Response:

```
HTTP/1.1 303 See Other
Content-Type: application/json
Location: /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"resourceReference": {
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001"
}}
```

D.13 Creating a chat message, using tel URI and returning the location of the created resource (section 6.7.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatMessage": {
   "reportRequest": "Displayed",
   "text": "How are you?"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location:
http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"resourceReference": {
  "resourceURL":
      "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001"
}}
```

D.14 Creating a chat message, using ACR and returning a copy of the created resource (section 6.7.5.2)

Request:

```
POST /exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/adhoc/messages HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatMessage": {
   "reportRequest": "Displayed",
   "text": "How are you?"
}}
```

Response:

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/adhoc/messages/msg001
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"chatMessage": {
  "reportRequest": "Displayed",
   "resourceURL":
       "http://example.com/exampleAPI/chat/v1/acr%3Apseudonym123/oneToOne/acr%3Apseudonym456/messages/adhoc/msg001",
  "text": "How are you?"
}}
```

D.15 Creating an “isComposing” message (section 6.7.5.3)

Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"isComposing": {
    "contenttype": "text/plain",
    "refresh": "90",
    "state": "active"
}}
```

Response

```
HTTP/1.1 201 Created
Content-Type: application/json
Location:
http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg002
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"resourceReference": {
   "resourceURL":
       "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg002"
}}
```

D.16 Creating a chat message during session set-up in Confirmed 1-1 Chat mode (see section 6.7.5.4)

Request

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/messages HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatMessage": {
    "reportRequest": "Displayed",
    "text": "How are you?"
}}
```

Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"requestError": {
```

```
   "link": {
      "href":
"http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/sess001/messages/msg002",
      "rel": "ChatMessage"
   },
   "policyException": {
      "messageId": "POL1012",
      "text": "Messages during session setup not supported."
   }
}}
```

D.17 Example: Reading the status of an individual message (section 6.8.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001/status HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"messageStatusReport": {"status": "Displayed"}}
```

D.18 Reporting the status of a chat message (section 6.8.4.1)

Request:

```
PUT /exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"messageStatusReport": {"status": "Displayed"}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.19 Creating a new group chat session (section 6.9.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group HTTP/1.1
Content-Length: nnnn
Content-Type: application/json
Accept: application/json
Host: example.com

{"groupChatSessionInformation": {
   "clientCorrelator": "12345",
   "participant": [
      {
         "address": "tel:+19585550100",
         "isOriginator": "true",
         "name": "Alice"
      },
      {
         "address": "tel:+19585550101",
         "name": "Bob"
      },
      {
         "address": "tel:+19585550102",
         "name": "Ted"
      }
   ],
   "subject": "Hello"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"groupChatSessionInformation": {
   "clientCorrelator": "12345",
   "participant": [
      {
         "address": "tel:+19585550100",
         "isOriginator": "true",
         "name": "Alice",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
         "status": "Connected"
      },
      {
         "address": "tel:+19585550101",
         "name": "Bob",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
         "status": "Invited"
      },
      {
         "address": "tel:+19585550102",
```

```
            "name": "Ted",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003",
            "status": "Invited"
        }
    ],
    "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001",
    "subject": "Hello"
}}
```


D.20 Retrieving group chat session information (section 6.10.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:


```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"groupChatSessionInformation": {
    "clientCorrelator": "12345",
    "participant": [
        {
            "address": "tel:+19585550100",
            "isOriginator": "true",
            "name": "Alice",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
            "status": "Connected"
        },
        {
            "address": "tel:+19585550101",
            "name": "Bob",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
            "status": "Invited"
        },
        {
            "address": "tel:+19585550102",
            "name": "Ted",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003",
            "status": "Invited"
        }
    ],
    "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001",
    "subject": "Hello"
}}
```

D.21 Retrieving group chat session information when being disconnected (section 6.10.3.2)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.22 Terminating a group chat session (section 6.10.6.1)

Request:

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.23 Retrieving the list of Participants in a group chat session (section 6.11.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"participantList": {
   "participant": [
      {
         "address": "tel:+19585550100",
         "isOriginator": "true",
         "name": "Alice",
```

```
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
            "status": "Connected"
        },
        {
            "address": "tel:+19585550101",
            "name": "Bob",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
            "status": "Connected"
        },
        {
            "address": "tel:+19585550102",
            "name": "Ted",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003",
            "status": "Connected"
        }
    ],
    "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants"
}}
```


D.24 Retrieving the list of Participants in a group chat session when being disconnected (section 6.11.3.2)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.25 Retrieving the list of Participants in a group chat session when not having access rights (section 6.11.3.3)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 403 Forbidden
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"requestError": {
    "link": {
```

```
      "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants",
      "rel": "ParticipantList"
   },
   "policyException": {
      "messageId": "POL1011",
      "text": "Access denied."
   }
}}
```

D.26 Adding one Participant to a group chat, or re-joining a group chat (section 6.11.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"participantInformation": {
   "address": "tel:+19585550103",
   "clientCorrelator": "12345",
   "name": "John"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"participantInformation": {
   "address": "tel:+19585550103",
   "clientCorrelator": "12345",
   "name": "John",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004",
   "status": "Invited"
}}
```

D.27 Adding multiple Participants to a group chat (section 6.11.5.2)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
```

```
Accept: application/json
Host: example.com

{"participantList": {
   "participant": [
      {
         "address": "tel:+19585550103",
         "clientCorrelator": "12345",
         "name": "John"
      },
      {
         "address": "tel:+19585550104",
         "clientCorrelator": "67890",
         "name": "Peter"
      }
   ]
}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"participantList": {
   "participant": [
      {
         "address": "tel:+19585550100",
         "isOriginator": "true",
         "name": "Alice",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
         "status": "Connected"
      },
      {
         "address": "tel:+19585550101",
         "name": "Bob",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
         "status": "Connected"
      },
      {
         "address": "tel:+19585550102",
         "name": "Ted",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003",
         "status": "Connected"
      },
      {
         "address": "tel:+19585550103",
         "clientCorrelator": "12345",
         "name": "John",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004",
         "status": "Invited"
      },
      {
         "address": "tel:+19585550104",
```

```
            "clientCorrelator": "67890",
            "name": "Peter",
            "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part005",
            "status": "Invited"
        }
    ],
    "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants"
}}
```


D.28 Error situation when trying to re-join a group chat session (section 6.11.5.3)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

{"requestError": {
    "link": {
        "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants",
        "rel": "ParticipantList"
    },
    "policyException": {
        "messageId": "POL1011",
        "text": "Access denied."
    }
}}
```

D.29 Retrieving information about an individual group chat Participant (section 6.12.3.1)

Request:

```
GET /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"participantInformation": {
   "address": "tel:+19585550103",
   "clientCorrelator": "12345",
   "name": "John",
   "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part004",
   "status": "Invited"
}}
```

D.30 Leaving a group chat session (section 6.12.6.1)

Request:

```
DELETE /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

D.31 Accepting a group chat invitation (section 6.13.4.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"participantSessionStatus": {"status": "Connected"}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"participantSessionStatus": {"status": "Connected"}}
```

D.32 Creating a group chat message, using tel URI and returning the location of the created resource (section 6.14.5.1)

Request:

```
POST /exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"chatMessage": {
   "reportRequest": "Displayed",
   "text": "How are you?"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"resourceReference": {
  "resourceURL": "http://example.com/exampleAPI/chat/v1/ tel%3A%2B19585550100/group/sess001/messages/msg001"
}}
```

D.33 Notify a client about incoming messages (section 6.15.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"messageNotification": {
   "callbackData": "abcd",
   "chatMessage": {
      "reportRequest": "Displayed",
      "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001",
      "text": "Hello Alice"
   },
   "dateTime": "2001-12-17T09:30:47Z",
   "link": {
      "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/messages/msg001",
      "rel": "ChatMessage"
   },
   "senderAddress": "tel:+19585550102",
   "senderName": "Ted"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.34 Notify a client about message status (section 6.16.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"messageStatusNotification": {
   "callbackData": "abcd",
   "link": {
      "href":
"http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/oneToOne/tel%3A%2B19585550101/adhoc/messages/msg001",
      "rel": "ChatMessage"
   },
   "status": "Displayed"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.35 Notify a client about 1-1 chat session invitations (section 6.17.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"sessionInvitationNotification": {
   "callbackData": "abcd",
   "link": {
      "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550101/oneToOne/tel%3A%2B19585550100/sess001",
      "rel": "ChatSessionInformation"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Alice",
   "subject": "Hi there!",
   "tParticipantAddress": "tel:+19585550101",
   "tParticipantName": "Bob"
```

```
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.36 Notify a client about group chat session invitations (section 6.18.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"groupSessionInvitationNotification": {
   "callbackData": "abcd",
   "link": [
      {
         "href": " http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550102/group/sess001",
         "rel": "GroupChatSessionInformation"
      },
      {
         "href": " http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550102/group/sess001/participants/part003",
         "rel": " ParticipantInformation"
      }
   ],
   "participant": [
      {
         "address": "tel:+19585550100",
         "isOriginator": "true",
         "name": "Alice",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
         "status": "Connected"
      },
      {
         "address": "tel:+19585550101",
         "name": "Bob",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
         "status": "Invited"
      },
      {
         "address": "tel:+19585550102",
         "name": "Ted",
         "resourceURL": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part003",
         "status": "Invited"
      }
   ],
   "subject": "Hello"
```

```
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.37 Notify a client about chat session events (section 6.19.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"chatEventNotification": {
   "callbackData": "abcd",
   "eventDescription": "The session has ended.",
   "eventType": "SessionEnded",
   "link": [
      {
         "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001",
         "rel": "GroupChatSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001",
         "rel": "ChatNotificationSubscription"
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.38 Notify a client about Participant status changes (section 6.20.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"participantStatusNotification": {
   "callbackData": "abcd",
```

```
  "link": [
    {
      "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001",
      "rel": "GroupChatSessionInformation"
    },
    {
      "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001",
      "rel": "ChatNotificationSubscription"
    }
  ],
  "participant": [
    {
      "address": "tel:+19585550100",
      "link": {
        "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part001",
        "rel": "ParticipantInformation"
      },
      "name": "Alice",
      "status": "Connected",
      "yourown": "true"
    },
    {
      "address": "tel:+19585550101",
      "link": {
        "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/group/sess001/participants/part002",
        "rel": "ParticipantInformation"
      },
      "name": "Bob",
      "status": "Disconnected",
      "yourown": "false"
    }
  ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.39 Notify a client about Participant status changes (section 6.21.5.1)

Request:

```
POST /chat/notifications/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: application.example.com

{"subscriptionCancellationNotification": {
  "callbackData": "abcd",
  "link": {
    "href": "http://example.com/exampleAPI/chat/v1/tel%3A%2B19585550100/subscriptions/sub001",
    "rel": "ChatNotificationSubscription"
```

```
  }
}}
```

7.2.4.1.1 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

Appendix E. Operations mapping to a pre-existing baseline specification (Informative)

As this specification does not have a baseline specification, this appendix is empty.

Appendix F. Light-weight resources (Informative)

As this version of the specification does not define any light-weight resources, this Appendix is empty.

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful Chat API in combination with some authorization frameworks.

G.1 Use with OMA Authorization Framework for Network APIs

The RESTful Chat API MAY support the authorization framework defined in [Autho4API_10].

A RESTful Chat API supporting [Autho4API_10]:

- SHALL conform to section D.1 of [REST_NetAPI_Common];
- SHALL conform to this section G.1.

G.1.1 Scope values

G.1.1.1 Definitions

In compliance with [Autho4API_10], an authorization server serving clients requests for getting authorized access to the resources exposed by the RESTful Chat API:

- SHALL support the scope values defined in the table below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_chat.all_{apiVersion}	Provide access to all defined operations on the resources in this version of the API. The {apiVersion} part of this identifier SHALL have the same value as the “apiVersion” URL variable which is defined in section 5.1. This scope value is the union of the other scope values listed in the next rows of this table.	No
oma_rest_chat.one_to_one	Provide access to all defined operations regarding 1-1 chats	No
oma_rest_chat.group	Provide access to all defined operations regarding groupchats	No

Table 1: Scope values for RESTful Chat API

G.1.1.2 Downscoping

In the case where the client requests authorization for “oma_rest_chat.all_{apiVersion}” scope, the authorization server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- “oma_rest_chat.one_to_one”
- “oma_rest_chat.group”

G.1.1.3 Mapping with resources and methods

Tables in this section specify how the scope values defined in section G.1.1.1 for the RESTful Chat API map to the REST resources and methods of this API. In these tables, the root "oma_rest_chat." of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to chat event notifications	/subscriptions	6.1	all_{apiVersion} or one_to_one or group	n/a	all_{apiVersion} or one_to_one or group	n/a
Individual subscription to chat event notifications	/subscriptions/{subscriptionId}	6.2	all_{apiVersion} or one_to_one or group	n/a	n/a	all_{apiVersion} or one_to_one or group

Table 2: Required scope values for: Subscriptions

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 chat sessions between two users	/{otherUserId}	6.3	n/a	n/a	all_{apiVersion} or one_to_one	n/a
Individual 1-1 chat session	/{otherUserId}/{sessionId}	6.4	all_{apiVersion} or one_to_one	n/a	n/a	all_{apiVersion} or one_to_one
1-1 chat session status	/{otherUserId}/{sessionId}/status	6.5	n/a	n/a	all_{apiVersion} or one_to_one	n/a
Extend 1-1 chat to a group chat session	/{otherUserId}/{sessionId}/extend	6.6	n/a	n/a	all_{apiVersion} or (one_to_one and group)	n/a
Chat messages in a 1-1 chat	/{otherUserId}/{sessionId}/messages	6.7	n/a	n/a	all_{apiVersion} or one_to_one	n/a

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}/oneToOne	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
Individual message status in a 1-1 chat	/{otherUserId}/{sessionId}/messages/{messageId}/status	6.8	all_{apiVersion} or one_to_one	all_{apiVersion} or one_to_one	n/a	n/a

Table 3: Required scope values for: 1-1 chats

Resource	URL Base URL: http://{serverRoot}/chat/{apiVersion}/{userId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
All group chat sessions	/group	6.9	n/a	n/a	all_{apiVersion} or group	n/a
Individual group chat session	/group/{sessionId}	6.10	all_{apiVersion} or group	n/a	n/a	all_{apiVersion} or group
All Participants in a group chat session	/group/{sessionId}/participants	6.11	all_{apiVersion} or group	n/a	all_{apiVersion} or group	n/a
Individual Participant in a group chat session	/group/{sessionId}/participants/{participantId}	6.12	all_{apiVersion} or group	n/a	n/a	all_{apiVersion} or group
Individual group chat session Participant status	/group/{sessionId}/participants/{participantId}/status	6.13	n/a	all_{apiVersion} or group	n/a	n/a
Chat messages in a group chat session	/group/{sessionId}/messages	6.14	n/a	n/a	all_{apiVersion} or group	n/a

Table 4: Required scope values for: Group chats

Resource	URL <specified by the client>	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification containing incoming	Specified by client when subscription is created or provisioned	6.15	n/a	n/a	all_{apiVersion} or one_to_one	n/a

Resource	URL <specified by the client>	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
message						
Client notification about message status	Specified by client when subscription is created or provisioned	6.16	n/a	n/a	all_{apiVersion} or one_to_one or group	n/a
Client notification about 1-1 chat session invitations	Specified by client when subscription is created or provisioned	6.17	n/a	n/a	all_{apiVersion} or one_to_one	n/a
Client notification about group chat session invitations	Specified by client when subscription is created or provisioned	6.18	n/a	n/a	all_{apiVersion} or group	n/a
Client notification about chat session events	Specified by client when subscription is created or provisioned	6.19	n/a	n/a	all_{apiVersion} or one_to_one or group	n/a
Client notification about changes of Participant status	Specified by client when subscription is created or provisioned	6.20	n/a	n/a	all_{apiVersion} or group	n/a
Client notification about subscription cancellation	Specified by client when subscription is created or provisioned	6.21	n/a	n/a	all_{apiVersion} or one_to_one or group	n/a

Table 5: Required scope values for: Notifications

Notifications are only sent to clients that have prior passed an authorization token with a matching scope in the related subscription.

G.1.2 Use of 'acr:Authorization'

This section specifies the use of 'acr:Authorization' in place of an end user identifier in a resource URL path.

An 'acr' URI of the form 'acr:Authorization', where 'Authorization' is a reserved keyword MAY be used to avoid exposing a real end user identifier in the resource URL path.

A client MAY use 'acr:Authorization' in a resource URL in place of the{userId} resource URL variable in the resource URL path, when the RESTful Chat API is used in combination with [Autho4API_10].

In the case the RESTful Chat API supports [Autho4API_10], the server:

- SHALL accept 'acr:Authorization' as a valid value for the resource URL variable {userId}.
- SHALL conform to [REST_Common_TS] section 5.8.1.1 regarding the processing of 'acr:Authorization'.