RESTful Network API for Messaging

Candidate Version 1.0 – 01 Dec 2015

Open Mobile Alliance

OMA-TS-REST_NetAPI_Messaging-V1_0-20151201-C

Contents

Figures

Tables

1. Scope

This specification defines a RESTful Messaging API using an HTTP protocol binding, based on the similar API defined in [3GPP 29.199-5].

2. References

2.1 Normative References

[3GPP 23.140]	3GPP Technical Specification, "Multimedia Messaging Service (MMS); Functional description; Stage 2 (Release 6)", URL:http://www.3gpp.org/
[3GPP 29.199-5]	3GPP Technical Specification, "Open Service Access (OSA); Parlay X Web Services; Part 5: Multimedia messaging (Release 8)", URL:http://www.3gpp.org/
[Autho4API_10]	"Authorization Framework for Network APIs", Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL:http://www.openmobilealliance.org/
[OMA_IM_TS]	"Instant Messaging using SIMPLE", Open Mobile Alliance™, OMA-TS-SIMPLE_IM-V1_0.doc, URL:http://www.openmobilealliance.org/
[REST_NetAPI_ACR]	"RESTful Network API for Anonymous Customer Reference Management", Open Mobile Alliance™, OMA-TS-REST_NetAPI_ACR-V1_0, 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_Messaging]	"XML schema for the RESTful Network API for Messaging", Open Mobile Alliance™, OMA-SUP-XSD_rest_netapi_messaging-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
[RFC2388]	"Returning Values from Forms: multipart/form-data" L.Masinter. August, 1998. URL:http://www.ietf.org/rfc/rfc2388.txt
[RFC3261]	"SIP: Session Initiation Protocol", J. Rosenberg et al., June 2002, URL:http://www.rfc-editor.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
[RFC4234]	"Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell. October 2005, URL:http://www.ietf.org/rfc/rfc4234.txt
[RFC7159]	"The JavaScript Object Notation (JSON) Data Interchange Format", T. Bray, Ed., March 2014, URL:http://tools.ietf.org/rfc/rfc7159.txt
[RFC7231]	"Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content," R. Fielding, Ed., J.Raschke, Ed., June 2014, URL: http://tools.ietf.org/html/rfc7231.txt
[SCRRULES]	"SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/
[W3C_FORMS]	"Use of Forms". URL:http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.2
[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
[WAP_SI]	Open Mobile Alliance. Service Indication for WAP Push messaging. URL:http://www.openmobilealliance.org/tech/affiliates/wap/wap-167-serviceind-20010731-a.pdf
[WAP_SL]	Open Mobile Alliance. Service Loading for WAP Push Messaging. URL:http://www.openmobilealliance.org/tech/affiliates/wap/wap-168-serviceload-20010731-a.pdf
[XMLSchema1]	W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures, W3C Recommendation 5 April 2012, URL:http://www.w3.org/TR/xmlschema11-1/

[XMLSchema2] W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes, W3C Recommendation 5 April 2012, URL:http://www.w3.org/TR/xmlschema11-2/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version 2.9, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL:http://www.openmobilealliance.org/

[ParlayREST_MMS] "RESTful bindings for Parlay X Web Services – Multi-media Messaging", Version 1.1, Open Mobile Alliance™, OMA-TS-ParlayREST_MultiMediaMessaging-V1_1 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/

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

For the purpose of this TS, all definitions from the OMA Dictionary apply [OMADICT].

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.
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.
Notification Server	A server that is capable of creating and maintaining Notification Channels.
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.

3.3 Abbreviations

ACR	Anonymous Customer Reference
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
EMS	Enhanced Message Service
GIF	Graphics Interchange Format
HTTP	HyperText Transfer Protocol
IM	Instant Message
ISDN	Integrated Services Digital Network
JPEG	Joint Photographic Expert Group
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
MMS	Multimedia Messaging Service
MSISDN	Mobile Subscriber ISDN Number
OMA	Open Mobile Alliance
PNG	Portable Network Graphics
REST	REpresentational State Transfer
RTX	Ring Tone eXtended
SCR	Static Conformance Requirements
SI	Service Indication
SIP	Session Initiation Protocol
SL	Service Loading

SMPP	Short Message Peer-to-Peer
SMS	Short Message Service
SMSC	Short Message Service Center
TS	Technical Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WAP	Wireless Application Protocol
WP	White Paper
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification for the OMA RESTful Network API for Messaging contains the HTTP protocol binding based on the Parlay X Multimedia Messaging Web Services [3GPP 29.199-5] specification, 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

The RESTful Network API for Messaging V1.0 is a republication of the ParlayREST MultiMediaMessaging API V1.1 [ParlayREST_MMS] as part of the suite of OMA RESTful Network APIs.
Bug fixes and structural changes to fit that suite, but also functional changes have been applied.

Version 1.0 of the RESTful Network API for Messaging keeps supporting the following operations:

- Send message to a terminal
- Check delivery status of the outgoing message
- Check incoming messages (polling mode)
- Create subscriptions for notifications for inbound messages based on given criteria (online)
- Delete subscriptions for notifications for inbound messages (online)
- Create subscriptions for notification for outbound messages based on given criteria (online)
- Delete subscriptions for notification for outbound messages (online)
- Retrieve message content
- Confirm message retrieval by deleting message (execute DELETE method)
- Report the status for an inbound message

The following new functionality has been introduced:

- 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:auth” as a reserved keyword in a resource URL variable that identifies an end user
- Support for the sub-structure “AttachmentInfo” within the structure “InboundMMSMessage”
- Support for voicemail message (which can contain voice, fax, or both in case of a fax with a voice comment).

All changes are backwards-compatible with ParlayREST MultiMediaMessaging V 1.1, with the following exceptions:

- the introduction of the sub-structure “AttachmentInfo” within the structure “InboundMMSMessage” is not backwards-compatible.

5. Messaging API definition

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

The terms “inbound” and “outbound” used in resource names and data structures refer to incoming, respectively outgoing messages from the client perspective. The term “subscription” refers to the online creation of resources (using requests in this specification). The term “registration” refers to the offline creation of resources using mechanisms out of scope of this specification. The resources created during registrations as well as subscriptions can generate notifications, for example about the delivery status of outgoing messages (subscription), or about incoming messages (registration).

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 5.3, 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, 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.

Section 7 contains fault definition details such as Service Exceptions and Policy Exceptions.

Appendix B provides the Static Conformance Requirements (SCR).

Appendix E lists the Parlay X equivalent operation for each supported REST resource and method combination, where applicable.

Appendix F provides a list of all light-weight 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 Resource summary

This section summarizes all the resources used by the RESTful Messaging API. The resources are defined with the goal of supporting unified messaging, to allow their re-use by other APIs.

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.

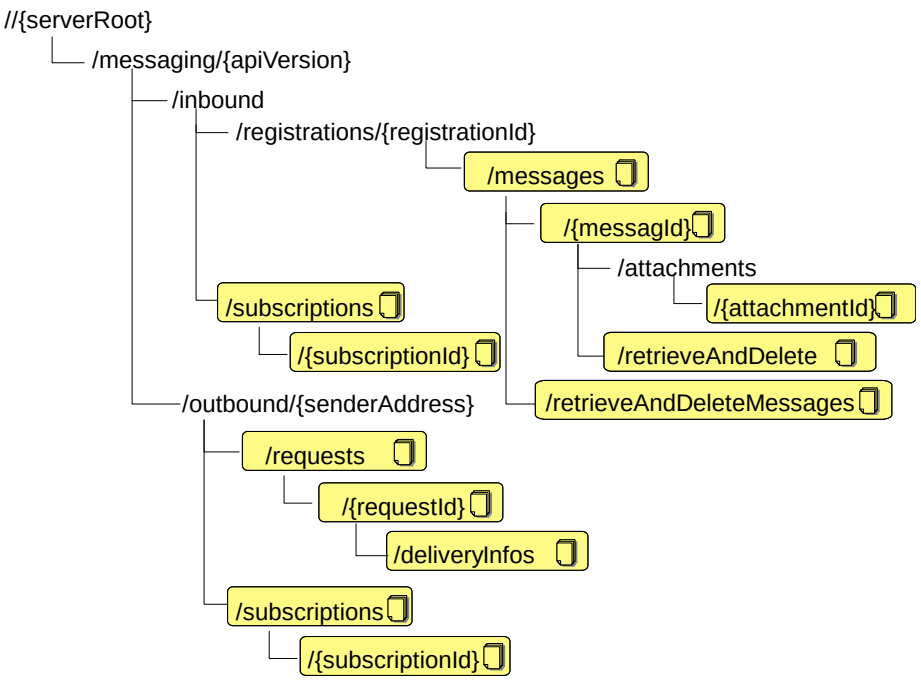

Figure 1 Resource structure defined by this specification

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.

Purpose: To allow client to periodically poll for inbound messages (based on provisioning step configuration)

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Inbound messages for a given registration	/inbound/registrations/{registrationId}/messages Note: Used by clients that periodically poll for incoming messages. Retrieval criteria have to be provisioned in advance.	InboundMessageList	read one or more messages from gateway storage	no	no	no
Inbound messages retrieve and delete using registration	/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages	InboundMessageList (used for POST response) InboundMessageRetrieveAndDeleteRequest (used for POST request)	no	no	pops one or more messages from the gateway storage (removes it if successful)	no

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Retrieval and deletion of individual inbound message using registration	/inbound/registrations/{registrationId}/messages/{messageId}/retrieveAndDelete	InboundMessage (used for POST response) InboundMessageRetrieveAndDeleteRequest (used for POST request)	no	no	pops one message and all attachments at once (mime formated) from the gateway storage (removes it if successful)	no
Inbound message for a given registration	/inbound/registrations/{registrationId}/messages/{messageId}	InboundMessage	read one message from gateway storage	no	no	delete one message from gateway storage
Inbound message attachment	/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId}	Any MIME content (the one of the attachment)	read individual message attachment	no	no	delete attach ment from gateway

Purpose: To allow client to manage subscriptions for inbound messages

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Inbound message subscriptions	/inbound/subscriptions	SubscriptionList (used for GET) Subscription (used for POST) common:ResourceReference (optional alternative for POST response)	read all active subscriptions	no	create new message subscription	no
Individual inbound message subscription	/inbound/subscriptions/{subscriptionId}	Subscription	read individual subscription	no	no	removes subscription and stops corresponding message notifications

Purpose: To allow server to notify client about inbound messages

Resource	URL <specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification about inbound message	<specified by the client when subscription is created or during provisioning process>	InboundMessageNotification	no	no	notifies client about new inbound message	no

Purpose: To allow client to inform server about the status for received inbound message

Resource	URL <specified by the server>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Individual inbound message status	<Specified by the server in the “link” element of an inbound message when the message is to be devivered to the receiving client>	MesageStatusReport	no	report the status of an inbound message on the receiver side	no	no

Purpose: To allow client to send messages and obtain delivery status for messages

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Outbound message requests	/outbound/{senderAddress}/requests	OutboundMessageRequestList (used for GET) OutboundMessageRequest (used for POST) common:ResourceReference (optional alternative for POST response)	read all pending outbound message reference with current delivery status	no	create new outbound messages request	no
Outbound message request and delivery status	/outbound/{senderAddress}/requests/{requestId}	OutboundMessageRequest	read message and delivery status for the individual outbound message request	no	no	no
Outbound message delivery status	/outbound/{senderAddress}/requests/{requestId}/deliveryInfos	DeliveryInfoList	read delivery status for the individual outbound message request	no	no	no

Purpose: To allow client to manage subscriptions for outbound message delivery status (overwrites individual request notifications)

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Outbound message delivery notification subscriptions	/outbound/{senderAddress}/subscriptions	DeliveryReceiptSubscriptionList (used for GET) DeliveryReceiptSubscription (used for POST) common:ResourceReference (optional alternative for POST response)	read all outbound message subscriptions	no	create new delivery receipt subscription	no
Individual outbound message delivery notification subscription	/outbound/{senderAddress}/subscriptions/{subscriptionId}	DeliveryReceiptSubscription	read an individual outbound message subscription	no	no	remove delivery recept notification subscription and stop corresponding delivery receipt notifications

Purpose: To allow server to notify client about outbound message delivery status

Resource	URL <specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Client notification about outbound message delivery status	<specified by the client when outbound request is submitted>	DeliveryInfoNotification	no	no	Notifies client about delivery status of outgoing requests	no

5.2 Data Types

5.2.1 XML Namespaces

The namespace for the Messaging data types is:

urn:oma:xml:rest:netapi:messaging:1

The 'xsd' namespace is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of the names 'xsd' and ‘common’ is not semantically significant.

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

Applications following the RESTful Network API for Messaging V 1.0 specification SHALL use the namespace urn:oma:xml:rest:netapi:messaging:1.

Note: Server implementations can choose to also support the legacy namespace urn:oma:xml:rest:messaging:1 for the Messaging data types, in order to allow backwards-compatibility with [ParlayREST_MMS] applications. Use of this legacy namespace is deprecated and support is foreseen to be withdrawn in future versions of this specification. In messages sent from the server to the application, the legacy namespace is suggested to be used by the server if it was used by a legacy application in the corresponding request or subscription message.

5.2.2 Structures

The subsections of this section define the data structures used in the RESTful Messaging 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: InboundMessageList

List of inbound messages

Element	Type	Optional	Description
inboundMessage	InboundMessage [0..unbounded]	Yes	It may contain an array of messages received according to the specified registrationid.
totalNumberOfPendingMessages	xsd:int	Yes	Total number of messages in the gateway storage waiting for retrieval at the time of the request
numberOfMessagesInThisBatch	xsd:int	Yes	Number of the messages included in the response (part of the totalNumberOfPendingMessages)
resourceURL	xsd:anyURI	No	Self referring URL

A root element named inboundMessageList of type InboundMessageList is allowed in response bodies.

When retrieving messages, in case the “inboundMessage” element contains the sub-element “reportRequest”, the recipient MUST acknowledge the requested event ‘Displayed’ as described in section 5.2.2.2.

5.2.2.2 Type: InboundMessage

Individual inbound message

Element	Type	Optional	Description
destinationAddress	xsd:anyURI	No	Number associated with the invoked messaging service, i.e. the destination address used by the terminal to send the message (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
senderAddress	xsd:anyURI	No	The address of the sender to whom a responding message may be sent (e.g.'sip' URI, 'tel' URI, 'acr' URI). If senderAddress is also part of the request URL, the two MUST have the same value.
dateTime	xsd:dateTimeStamp	Yes	Time when message was received by server
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.
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship with the resource, If reportRequest element is present, the server MUST include link to the resource containing message status which the receiving client SHALL use to report the status of the message (e.g. Displayed). The resource URL of that resource SHALL be included in the “href” attribute of the link element with rel=”MessageStatusReport”.

messageId	xsd:string	Yes	Server generated message identifier. This field MUST be present when the type of the message differs from a plain text SMS, i.e. the element in the choice below has a type other than InboundSMSTextMessage.
reportRequest	DeliveryStatus [0…unbounded]	Yes	List of delivery status events to report
inboundSMSTextMessage	InboundSMSTextMessage	Choice	Inbound SMS text message
inboundMMSMessage	InboundMMSMessage	Choice	Inbound MMS message
inboundIMMessage	InboundIMMessage	Choice	Inbound IM message
inboundVMMessage	InboundVMMessage	Choice	Inbound voicemail message

XSD modelling uses a “choice” to select either inboundSMSTextMessage, or inboundMMSMessage, or inboundIMMessage.

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

When retrieving a message, in case the root element “inboundMessage” 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”. See section 6.15 for more details.

5.2.2.3 Type: InboundMessageNotification

Notification about an individual inbound 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 inbound messages. See [REST_NetAPI_Common] for details.
inboundMessage	InboundMessage	No	Inbound message
link	common:Link[0..unbounded]	Yes	Link to other resources. For example a link to the subscription used to receive this message.

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

In case the “inboundMessage” element contains the sub-element “reportRequest”, the recipient MUST acknowledge the requested event ‘Displayed’ as described in sections 5.2.2.2.

5.2.2.4 Type: InboundSMSTextMessage

Content of an inbound textual SMS message

Element	Type	Optional	Description
message	xsd:string	No	Short message content

5.2.2.5 Type: InboundMMSMessage

Parameters for an inbound MMS message

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the received message.
priority	MessagePriority	Yes	The priority of the message: default is Normal.
attachment	AttachmentInfo [0…unbounded]	Yes	Information about individual attachments, including content type indication, the link for individual attachment retrieval and optionally the size of the attachment. In case the message contains a presentation part, this SHALL be referenced by the first item in the list of attachment elements.
bodyText	xsd:string	Yes	Contains the message body if it is encoded as ASCII text.

5.2.2.6 Type: InboundIMMessage

Parameters for an inbound IM message

Element name	Element type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the received IM message.
priority	MessagePriority	Yes	The priority of the message: default is Normal.
attachment	AttachmentInfo [0…unbounded]	Yes	Information about individual attachments, including content type indication, the link for individual attachment retrieval and optionally the size of the attachment. In case the message contains a presentation part, this SHALL be referenced by the first item in the list of attachment elements.
imFormat	IMFormat	Yes	If present, indicates the type of the received IM message. Otherwise any IM message type could be assumed (for example, server could not determine what type the received IM message is).
bodyText	xsd:string	Yes	Contains the message body if it is encoded as ASCII text.

5.2.2.7 Type: InboundVMMessage

A voicemail message can contain voice (i.e. audio media type), fax (i.e. image or document media type), or both in case of a fax with a voice comment (i.e. more than one media type).

Parameters for an inbound voicemail message.

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the received message.
duration	xsd:int	Yes	Voicemail message duration (in seconds).
numberOfFaxPages	xsd:int	Yes	Number of voicemail fax pages.
voiceTranscript	xsd:string	Yes	A text representation of the received voice.
priority	MessagePriority	Yes	The priority of the message: default is Normal.
attachment	AttachmentInfo [0…unbounded]	Yes	Information about individual attachments, including content type indication, the link for individual attachment retrieval and optionally the size of the attachment.

5.2.2.8 Type: SubscriptionList

List of subscriptions to notifications about inbound messages

Element	Type	Optional	Description
subscription	Subscription[0..unbounded]	Yes	It may contain an array of Subscription.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named subscriptionList of type SubscriptionList is allowed in response bodies

5.2.2.9 Type: Subscription

Individual subscription to notifications about inbound messages

Element	Type	Optional	Description
callbackReference	common:CallbackReference	No	Client's notification endpoint and parameters
destinationAddress	xsd:anyURI [1…unbounded]	No	The destination address of the message (e.g. 'sip' URI, 'tel' , 'acr' URI)
criteria	xsd:string	Yes	The text to match against to determine the application to receive the notification. This text is matched against the first word, defined as the initial characters after discarding any leading whitespace and ending with a whitespace or end of the string. The matching SHALL be case-insensitive. If the subject of the message is present it SHALL be used as the string, if not the string is defined as the first plain/text part of the content [3GPP 23.140]
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 subscription creation 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.
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship with the resource
useAttachmentURLs	xsd:boolean	Yes	Default: false. If set to 'true', inbound message has links to attachments together with the indication of the content type and optionally the size of each attachment. Otherwise, inbound message includes attachments using MIME.

A root element named subscription of type Subscription 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 specification [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

5.2.2.10 Type: InboundMessageRetrieveAndDeleteRequest

Parameters of the request to retrieve and delete messages in one operation

Element	Type	Optional	Description
retrievalOrder	RetrievalOrder	Yes	Specifies order in which messages should be retrieved if there are more than one pending.
priority	MessagePriority	Yes	The priority of the message: default is Normal.
maxBatchSize	xsd:int	Yes	Specifies maximum number of messages to be returned in the response.
useAttachmentURLs	xsd:boolean	No	If set to 'true', inbound message will have links to attachments together with the indication of the content type and optionally the size of each attachment. Otherwise, inbound message includes attachments using MIME.

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

5.2.2.11 Type: OutboundMessageRequestList

List of outbound message requests

Element	Type	Optional	Description
outboundMessageRequest	OutboundMessageRequest [0..unbounded]	Yes	Outbound message requests that have been sent by the application and still exist in the server. Message requests usually exist on the server for a little time after reaching their final delivery status.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named outboundMessageRequestList of type OutboundMessageRequestList is allowed in response bodies.

5.2.2.12 Type: OutboundMessageRequest

Individual outbound message request

Element	Type	Optional	Description
address	xsd:anyURI [1..unbounded]	No	Destination addresses for the Message (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
senderAddress	xsd:anyURI	No	The address of the sender to whom a responding message may be sent (e.g. 'sip' URI, 'tel' URI, 'acr' URI). If senderAddress is also part of the request URL, the two MUST have the same value.
senderName	xsd:string	Yes	Name of the sender to appear on the user's terminal as the originator of the message. If this parameter is used, a set of allowed values are assumed to be set during the provisioning of each sender (i.e.: for each user provisioned in the system).
charging	common:Charging Information	Yes	Charging to apply to this message
receiptRequest	common:CallbackReference	Yes	It defines the notification endpoint and parameters that will be used to notify the application when the message has been delivered to terminal or if delivery is impossible.
reportRequest	DeliveryStatus [0…unbounded]	Yes	List of delivery status events to report
outboundSMSTextMessage	OutboundSMSTextMessage	Choice	Included if a SMSText is being sent
outboundSMSBinaryMessage	OutboundSMSBinaryMessage	Choice	Included if a SMS Binary is being sent
outboundSMSLogoMessage	OutboundSMSLogoMessage	Choice	Included if a SMSLogo is being sent
outboundSMSRingToneMessage	OutboundSMSRingToneMessage	Choice	Included if a SMSRingtone is being sent
outboundSMSFlashMessage	OutboundSMSFlashMessage	Choice	Included if a Flash SMS is being sent
outboundWAPMessage	OutboundWAPMessage	Choice	Included if WAP is being used
outboundMMSMessage	OutboundMMSMessage	Choice	Included if MMS is being sent

outboundIMMessage	OutboundIMMessage	Choice	Included if IM is being sent
outboundVMMessage	OutboundVMMessage	Choice	Included if voicemail is being sent (i.e. deposit to storage and send notifications to device(s)).
outboundMessage	OutboundMessage	Choice	Included if the application does not care about which bearer is used to send a multimedia or text message.
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 field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate outbound message request creation in such situations. In case the field 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.
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship with the resource
deliveryInfoList	DeliveryInfoList	Yes	The delivery Information (filled in by the server)

XSD modelling uses a “choice” to select outboundSMSTextMessage, outboundSMSBinaryMessage, outboundSMSLogoMesage, outboundSMSRingToneMessage, outboundSMSFlashMessage, outboundWAPMessage, outboundMMSMessage, outboundIMMessage, outboundVMMessage or outboundMessage.

Note: outboundSMSBinaryMessage is supported in order to facilitate legacy applications that may send SMS in binary format (e.g. using SMPP). Underlying implementations need to be aware whether SMSCs and/or final destination mobile phones can handle such messages without unforeseen side effects. Implementations MUST support Service Provider policies to accept or reject the handling of a binary SMS message (POL1019: Policy error SHALL be used in case the message is rejected, see section 7.2).

A root element named outboundMessageRequest of type OutboundMessageRequest is allowed in request and/or response bodies

Regarding the clientCorrelator field, the note in section 5.2.2.9 applies.

5.2.2.13 Type: OutboundMMSMessage

Parameters of an outbound message

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the message.
priority	MessagePriority	Yes	The priority of the message: default is Normal.

5.2.2.14 Type: OutboundWAPMessage

Parameters of an outbound WAP message

Element	Type	Optional	Description
contentType	WAPContent	No	The type of content delivery notification to send
targetURL	xsd:anyURI	No	A URL from which content may be loaded by a terminal
serviceLoadingAction	ServiceLoadingAction	Choice	There is no user intervention. If the parameter is not specified, the default value will be “ExecuteLow”. See [WAP_SL] for more details. May be present only if ContentType is “ServiceLoading”.
serviceIndicationAction	ServiceIndicationAction	Choice	Allows controlling the level of intrusiveness, of outbound WAP push messages. According to [WAP_SI] it contains a text string specifying the action to be taken when the message is received. If the parameter is not specified, the value "SignalMedium" is used. May be resent only if ContentType is “ServiceIndicatiion”.
text	xsd:string	Yes	Information that accompanies the push. May be present only if ServiceIndicationAction is present and ContentType is “ServiceIndicatiion”.
created	xsd:dateTimeStamp	Yes	This attribute may be used to specify the date and time

			associated with the creation or last modification of the content indicated by targetURL, which may differ from the date and time when the message was created. May be present only if ContentType is "ServiceIndicatiion".

XSD modelling uses a "choice" to select either a serviceLoadingAction or serviceIndicationAction plus text and created.

5.2.2.15 Type: OutboundSMSTextMessage

Content of an outbound textual SMS message

Element	Type	Optional	Description
message	xsd:string	No	Short message content

5.2.2.16 Type: OutboundSMSBinaryMessage

Content of an outbound binary SMS message

Element	Type	Optional	Description
message	xsd:base64Binary	No	Short message content in binary format

5.2.2.17 Type: OutboundSMSLogoMessage

Content of an outbound SMS Logo message

Element	Type	Optional	Description
image	xsd:base64Binary	No	The image in jpeg, gif or png format. The image will be scaled to the proper format.
smsFormat	SmsFormat	No	Conversion to be applied to the message prior to delivery. Possible values are: 'Ems' or 'SmartMessaging'.

5.2.2.18 Type: OutboundSMSRingToneMessage

Content of an outbound SMS Ringtone message

Element	Type	Optional	Description
ringTone	xsd:string	No	The ring-tone in RTX format. Note:In the RTX Ringtone Specification, an RTX file is a text file containing the ring-tone name, a control subclause and a subclause containing a comma separated sequence of ring tone commands.
smsFormat	SmsFormat	No	Conversion to be applied to the message prior to delivery. Possible values are: 'Ems' or 'SmartMessaging'.

5.2.2.19 Type: OutboundSMSFlashMessage

Content of an outbound Flash SMS message.

Element	Type	Optional	Description
flashMessage	xsd:string	No	Content of Flash message

5.2.2.20 Type: OutboundIMMessage

Parameters of an outbound IM message

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the received message.
imFormat	IMFormat	Yes	The type of IM
bodyText	xsd:string	Yes	Contains the message body if it is encoded as ASCII text.

5.2.2.21 Type: OutboundVMMessage

A voicemail message can contain voice (i.e. audio media type), fax (i.e. image or document media type), or both in case of a fax with a voice comment (i.e. more than one media type).

The format of the voicemail message is according to its MIME Content-Type.

Parameters for an outbound voicemail message.

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the sent message.
duration	xsd:unsignedInt	Yes	Voicemail message duration (in seconds).
numberOfFaxPages	xsd:unsignedInt	Yes	Number of voicemail fax pages.
voiceTranscript	xsd:string	Yes	A text representation of the outgoing voice.
priority	MessagePriority	Yes	The priority of the message: default is Normal.

5.2.2.22 Type: OutboundMessage

Parameters of a generic outbound message (i.e. a bearer-agnostic outbound message). This message type can be used by the application to send text or multimedia messages when the application does not care about the bearer via which the message is delivered. Text messages are typically inlined in the "message" field. Multimedia messages are multipart constructs as defined in section 5.2.5.

Element	Type	Optional	Description
subject	xsd:string	Yes	If present, indicates the subject of the message. Not relevant for text messages.
priority	MessagePriority	Yes	The priority of the message: default is Normal.
message	xsd:string	Yes	Text of the message. Only relevant for text messages.

5.2.2.23 Type: DeliveryInfoList

List of delivery information records for an outbound message request

Element	Type	Optional	Description
resourceURL	xsd:anyURI	No	Self referring URL
link	common:Link[0..unbounded]	Yes	Linked to other resources that are in relationship with the resource
deliveryInfo	DeliveryInfo[1…unbounded]	No	Delivery information

A root element named deliveryInfoList of type DeliveryInfoList is allowed in response bodies.

5.2.2.24 Type: DeliveryInfoNotification

Notification about changes in the delivery information of an outbound message request

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'receiptRequest' element when creating an outbound message request. See [REST_NetAPI_Common] for details.
deliveryInfo	DeliveryInfo [1..unbounded]	No	Delivery information
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship to the notification. For example a link to the original outbound message request. The server MAY also include a link to the related subscription to notifications.

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

5.2.2.25 Type: DeliveryInfo

Delivery information of an outbound message request regarding one recipient address

Element	Type	Optional	Description
address	xsd:anyURI	No	Outbound message destination address (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
deliveryStatus	DeliveryStatus	No	Indicates the delivery result for the destination address.
description	xsd:string	Yes	Used together with delivery status (e.g.DeliveryImpossible) to provide additional information
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship to the notification. For example a link to the original outbound message request.

5.2.2.26 Type: DeliveryReceiptSubscriptionList

List of subscriptions to notifications about changes in the delivery information of an outbound message request

Element	Type	Optional	Description
resourceURL	xsd:anyURI	No	Self referring URL
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship with the resource
deliveryReceiptSubscription	DeliveryReceiptSubscription[0…unbounded]	Yes	Delivery information

A root element named deliveryReceiptSubscriptionList of type DeliveryReceiptSubscriptionList is allowed in response bodies.

5.2.2.27 Type: DeliveryReceiptSubscription

Individual subscription to notifications about changes in the delivery information of an outbound message request

Element	Type	Optional	Description
callbackReference	common:CallbackReference	No	Notification endpoint and parameters definition
filterCriteria	xsd:string	No	The filterCriteria will allow the service to filter flexibly. One example would be for the Service Provider to filter based on first 4 digits in MSISDN. This however is implementation specific and will be left to the Service Provider.
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 subscription creation 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.
link	common:Link[0..unbounded]	Yes	Link to other resources that are in relationship with the resource

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

Regarding the clientCorrelator field, the note in section 5.2.2.9 applies.

5.2.2.28 Type: AttachmentInfo

Parameters for an inbound message

Element	Type	Optional	Description
contentType	xsd:string	No	Indicates the content type of the attachment. For example: image/gif, video/3gpp
size	xsd:unsignedLong	Yes	Indicates the actual size of the original attachment in bytes.
link	common:Link	No	Link to individual attachment: E.g.: <link rel=”attachment” href=”../inbound/registration/{registrationId}/messages/{messageId}/attachments/{attachmentId}”/>)

5.2.2.29 MessageStatusReport

This type represents a response to the inbound message retrieval.

It is only needed if the inbound message includes an indication that the sender wants to receive a report about “Displayed” message status.

Note that the report regarding the “DeliveredToTerminal” 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	DeliveryStatus	No	Indicates the status of the message

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

5.2.3 Enumerations

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

5.2.3.1 Enumeration: DeliveryStatus

Delivery status enumeration

Enumeration	Description
DeliveredToTerminal	Successful delivery to receiver's terminal
DeliveryUncertain	Delivery status unknown: e.g. because it was handed off to another network
DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.
DeliveredToNetwork	Successful delivery to the network entity responsible for distributing the message further in the network
DeliveryNotificationNotSupported	Unable to provide delivery receipt notification. NotifyMessageDeliveryReceipt function will provide DeliveryNotificationNotSupported to indicate that delivery receipt for the specified address in a send message is not supported.
Displayed	The message is displayed on the receiver's terminal

5.2.3.2 Enumeration: IMFormat

IM format enumeration

Enumeration	Description
IM	Instant (immediate) messaging service (Can be short IM or large IM. Underlying network can decide message type from message context)
IMPagerMode	Short IM text message, as defined in [OMA_IM_TS]
IMLargeMode	Large IM message, as defined in [OMA_IM_TS]
IMFileTransfer	Large IM used for File Transfer, as defined in [OMA-IM_TS]

5.2.3.3 Enumeration: MessagePriority

Message priority enumeration

Enumeration	Description
Default	Default message priority
Low	Low message priority
Normal	Normal message priority
High	High message priority

5.2.3.4 Enumeration: RetrievalOrder

Retrieval order enumeration

Enumeration	Description
OldestFirst	Retrieve in the order from oldest to newest
NewestFirst	Retrieve in the order from newest to oldest

5.2.3.5 Enumeration: ServiceIndicationAction

Service indication enumeration for WAP messages

Enumeration	Description
SignalNone	The message MUST NOT be presented or postponed. If anything, only the info part could be used by the client for some purpose [WAP_SI].
SignalLow	The SI MUST be postponed without user intervention
SignalMedium	The SI MUST be presented as soon as the implementation allows that to be carried out in a non-user-intrusive manner.
SignalHigh	The SI MUST be presented as soon as the implementation allows that to be carried out in a non-user-intrusive manner, or earlier if considered appropriate (which MAY result in a user-intrusive behaviour). This decision can either be based on user preference settings or be carried out at the discretion of the implementation.
Delete	The message should be discarded.

5.2.3.6 Enumeration: ServiceLoadingAction

Service loading enumeration for WAP messages

Enumeration	Description
ExecuteLow	The service identified by the URI provided by the SL's href attribute is loaded in the same way as the user agent otherwise performs method requests initiated by the end-user. This implies that service content is fetched either from an origin server or from the client's cache, if available. Once the method request is successfully completed, the user agent loads the service into a clean user agent context and executes it. This MUST be carried out in an non-user-intrusive manner [WAP_SL]
ExecuteHigh	The service is loaded and executed in the same way as for ExecuteLow, but MAY result in a user-intrusive behavior.
Cache	The service is loaded in the same way as for ExecuteLow. However, instead of executing the service (as described above) it is placed in the cache of the client. If no cache exists, the SL MUST be silently discarded.

5.2.3.7 Enumeration: SmsFormat

SMS format enumeration

Enumeration	Description
Ems	EMS conversion
SmartMessaging	SmartMessaging® conversion

5.2.3.8 Enumeration: WAPContent

WAP content enumeration

Enumeration	Description
ServiceIndication	The Service Indication (SI) content type provides the ability to send notifications to end-users in an asynchronous manner. In its most basic form, an SI contains a short message and a URI indicating a service. The message is presented to the end-user upon reception, and the user is given the choice to either start the service indicated by the URI immediately, or postpone the SI for later handling. [WAP_SI].
ServiceLoading	The Service Loading (SL) content type provides the ability to cause a user agent on a mobile client to load and execute a service. The SL contains a URI indicating the service to be loaded by the user agent without user intervention when appropriate. [WAP_SL].

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):

- InboundMessage
- InboundMessageList
- Subscription
- SubscriptionList
- MessageStatusReport
- OutboundMessageRequest
- OutboundMessageRequestList
- DeliveryInfoList
- DeliveryReceiptSubscription
- DeliveryReceiptSubscriptionList
- attachment

These values indicate the kind of resource that the link points to. The value "attachment" indicates that the Link refers to an attachment of the message.

5.2.5 MIME multipart representation

MMS messages (inbound and outbound) and Generic messages (outbound) can be represented as multipart/form-data entity bodies, where the first entry of the form are the root fields and the second entry of the form are the attachments. Details about the structure of such messages are defined in [REST_NetAPI_Common]. The type of the form entry carrying the root fields part of such a message MUST be either InboundMessage or OutboundMessageRequest in this API. In case the MMS message has a presentation part, this part SHALL be the first MIME message body part after the root part, i.e. the first part of the multipart/mixed body.

Note: An inbound message can alternatively be represented as a list of link elements to the individual attachments.

5.3 Sequence Diagrams

The following subsections describe the resources, methods and steps involved in typical scenarios.

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 Send message and check the delivery status

This figure below shows a scenario for sending a message and get the delivery status of the message.

The resources:

– To send a message, create new resource under

http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests

– To get the delivery status of the message, do either a. or b.

a. read the newly created resource including the delivery status of the message

http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests/{requestId}

b. directly read the resource

http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests/{requestId}
/deliveryInfos

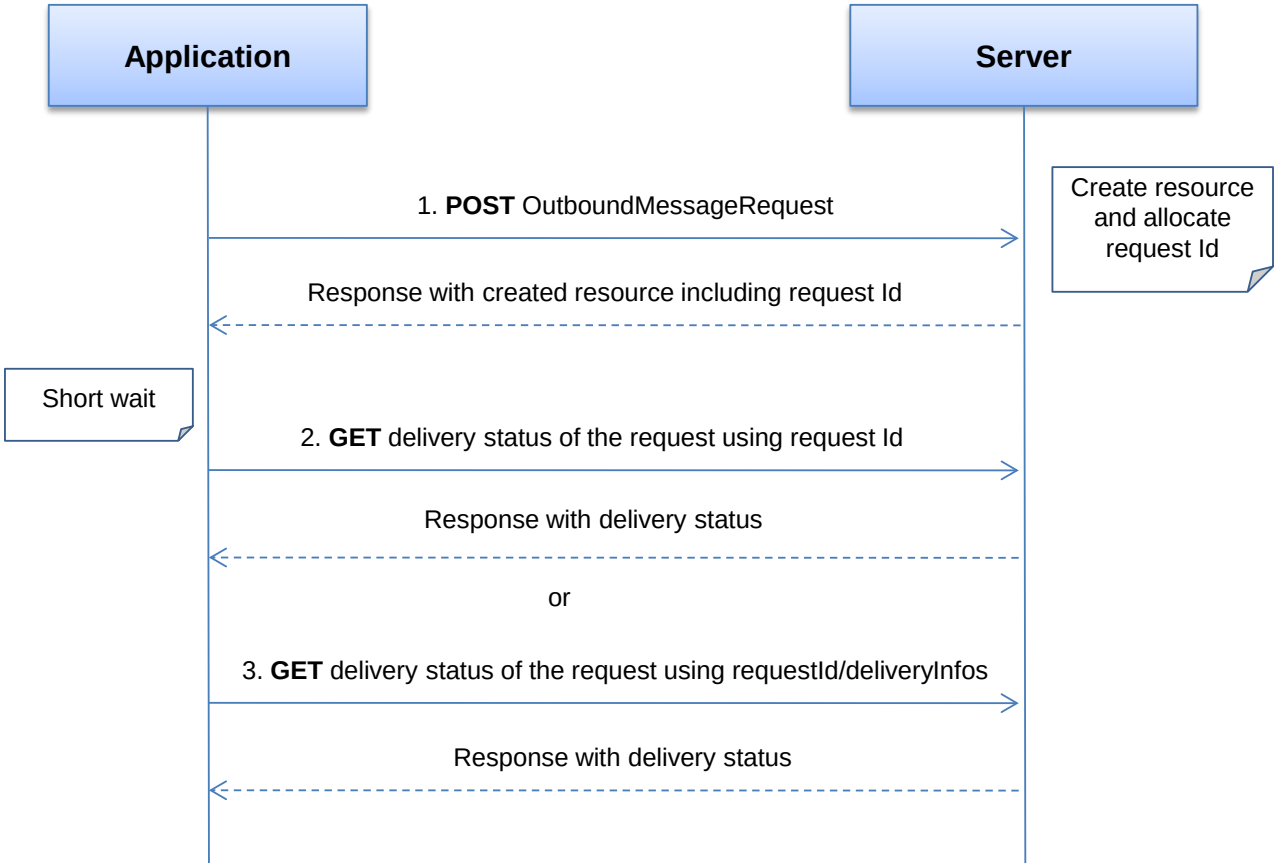

Figure 2 Send message and check the delivery status

Outline of the flows:

1. An application initiates the creation of new outbound message request using POST and receives the created resource with a resource URL containing the requestId.
2. The application requests the resource of the sent message with the given resource URL (containing the requestId) using GET and optionally gets the delivery status, or
3. The application requests the delivery status of the sent message with the given delivery info list URL using GET and gets the status.

5.3.2 Inbound message delivery (push mode)

This figure below shows a scenario for starting notification of inbound message with specific criteria on-line and receiving it when the message having the specified criteria arrives.

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 the use of that Notification Channel by the client.

The resources:

– To start subscription to notifications for inbound messages, create new resource under

http://{serverRoot} /messaging/{apiVersion}/inbound/subscriptions

– (The message is received in a notification)

– To update message status (notify server that the message has been displayed), update the resource containing message status with the resource URL received with the inbound message in the link element.

– To stop the subscription to notifications, delete the resource

http://{serverRoot} /messaging/{apiVersion}/inbound/subscriptions/{subscriptionId}

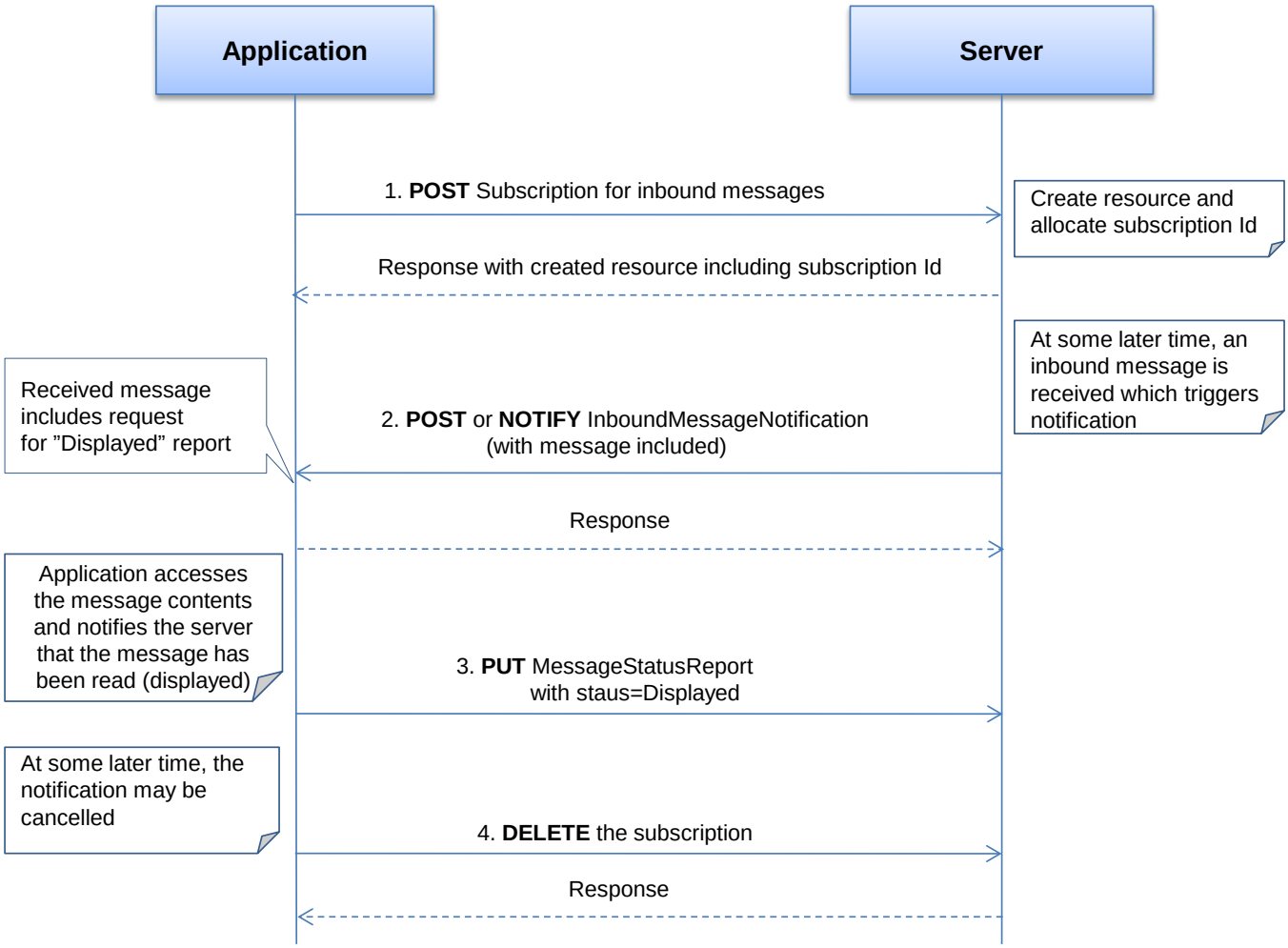

Figure 3 Inbound message delivery (push mode)

Outline of the flows:

1. An application subscribes to notifications for inbound messages using POST method and receives the resulting resource URL containing the subscriptionId.
2. When the message which satisfies the specified criteria arrives, the REST service on the server notifies the application of the message arrival using POST so that the application may read the message request. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].
3. The received message includes request for "Displayed" message status report and the application using PUT method on the resource containing message status updates the message status to "Displayed". If the message does not include request for "Displayed" message status report, this step SHOULD be skipped.
4. The application stops the notifications subscription using DELETE with a resource URL containing the subscriptionId.

5.3.3 Inbound message delivery (polling mode)

This figure below shows a scenario for checking for incoming messages using retrieval criteria that is set up offline ,getting one message ,and deleting it from the storage.

The resources:

- To retrieve incoming messages satisfying the criteria set up in advance, get the resource

  http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages

  This will return message references (identifiers and if requested, attachments URLs).

- To read one message from the storage, get the resource

  http://{serverRoot} /messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}

  This will return the whole message (MIME format)

- To read individual attachments of an message, based on message identifiers and attachment URLs:

  http://{serverRoot}/{apiVersion}
  /messaging/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId}

- To update message status (notify server that the message has been displayed), update the resource containing message status with resource URL received with inbound message in the link element.

- To remove one message from the storage, delete the resource

  http://{serverRoot} /messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}

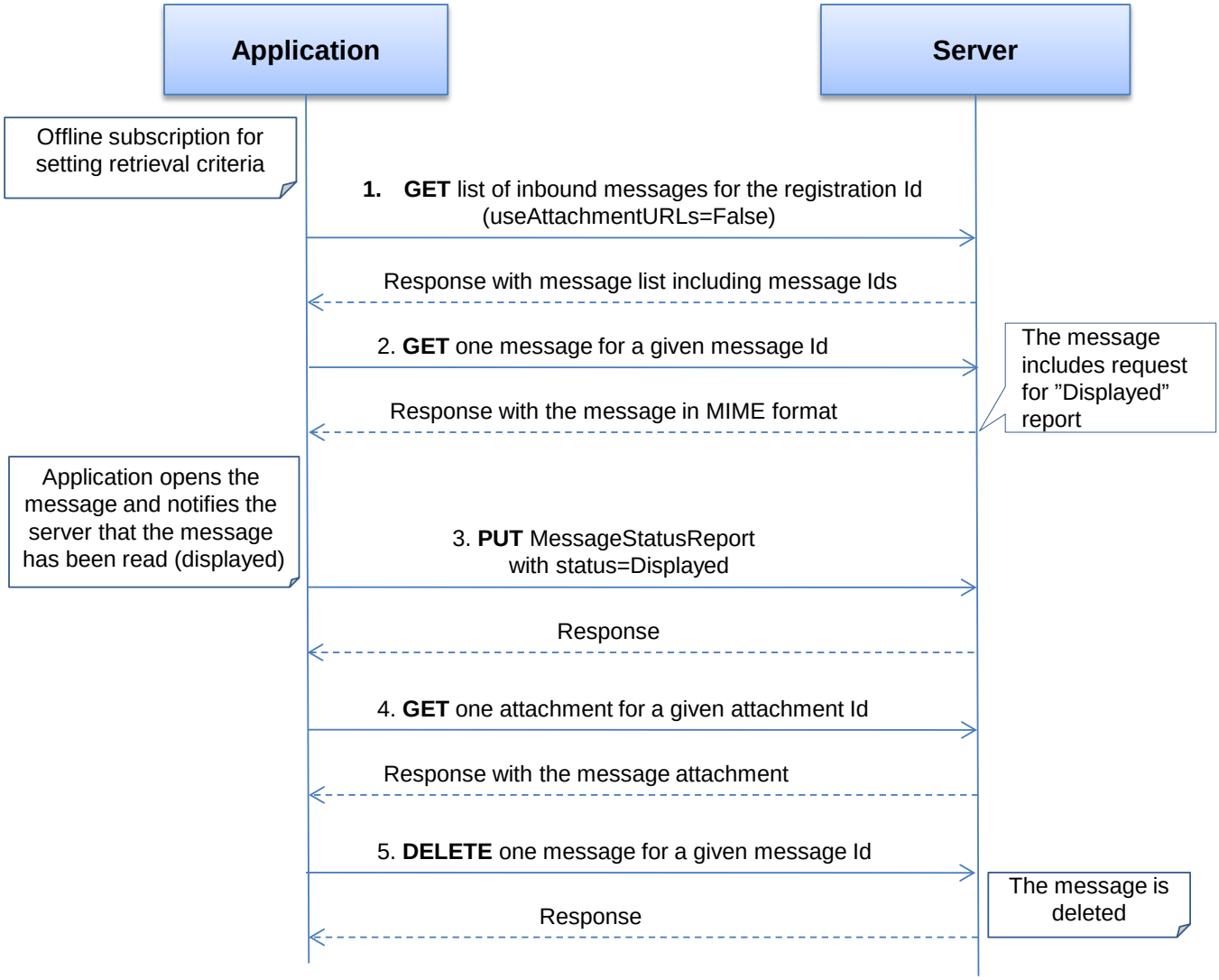

Figure 4 Inbound message delivery (polling mode)

Outline of the flows:

1. In advance, the notification of message reception with specific criteria is registered offline. An application requests the list of the incoming messages fulfilling specified criteria using GET with a resource URL containing the registrationId.
2. The application reads one message using GET method with a resource URL containing the messageId.
3. The received message includes request for “Displayed” report and the application using PUT method on the resource containing message status updates the message status to “Displayed”. Note that if the message does not include request for “Displayed” report, this step SHOULD be skipped.
4. The application reads one attachment to the message using GET with a resource URL containing the attachmentId.

The application removes one of the messages from gateway storage using DELETE with a resource URL containing the messageId.

5.3.4 Subscription to notifications for outbound message delivery status

This figure below shows a scenario for starting notifications for outbound messages delivery status.

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 the use of that Notification Channel by the client.

The resources:

- To start subscription to notifications for outbund messages delivery status, create new resource under

  http://{serverRoot} /messaging/{apiVersion}/outbound/{senderAddress}/subscriptions

- To notify the application about the message delivery status, POST notification to the client supplied notification URL
- To stop the subscription to notifications, delete the resource

  http://{serverRoot} /messaging/{apiVersion}/outbound/{senderAddress}/subscriptions/{subscriptionId}

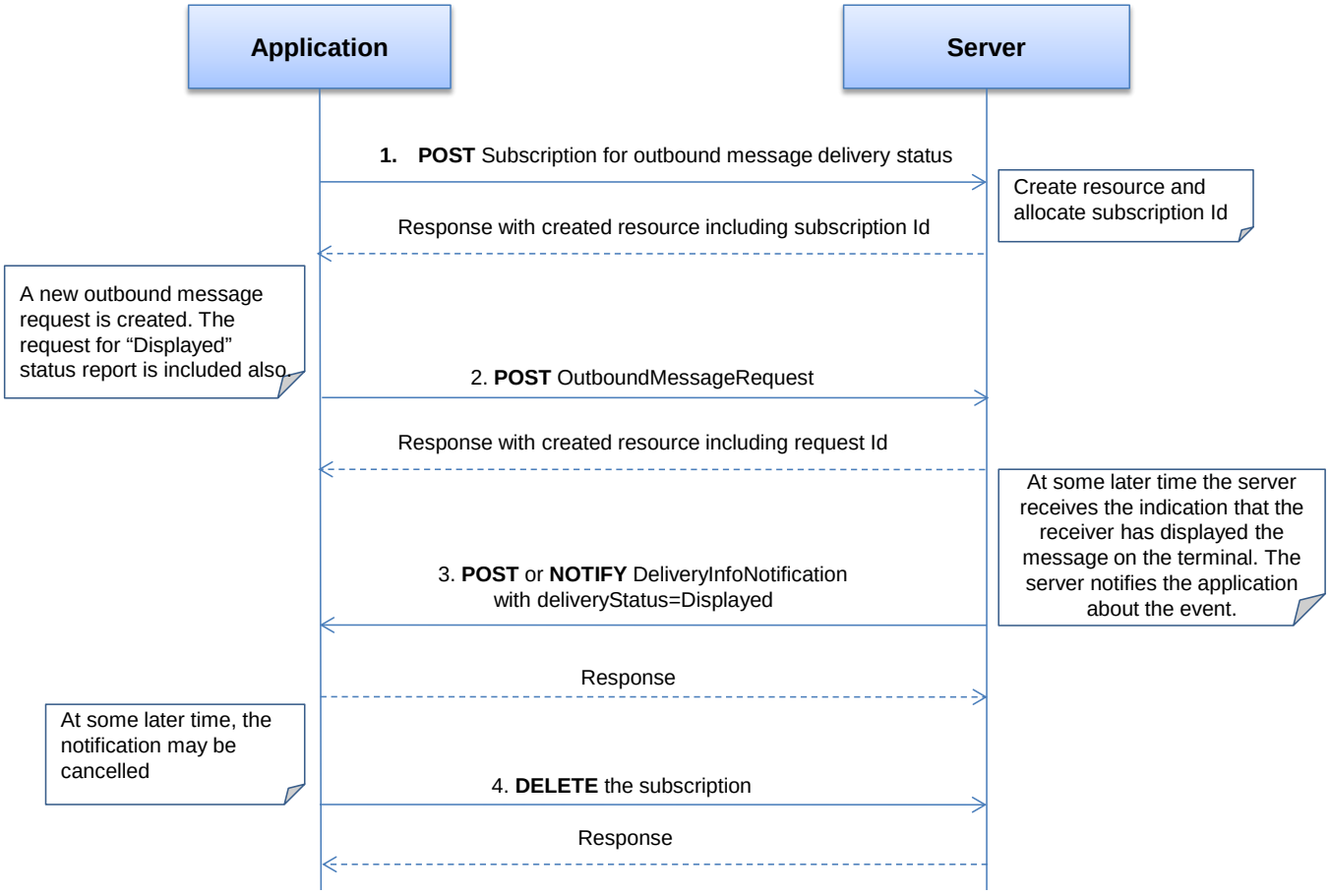

Figure 5 Subscription to notifications on outbound message delivery status

Outline of the flows:

1. An application subscribes to notifications for outbound messages delivery status by using the POST method, and receives the resulting resource URL containing the subscriptionId.
2. Later the application decides to send a new outbound message request by using POST method as described in 5.3.1. The request includes also an indication that the application wants to be notified when the message is displayed on the receiver's terminal. The response included created resource with requestId.
3. When the message is displayed on the receiver's terminal, the REST service on the server notifies the application about the event by using POST method. Alternatively, the application obtains the notification using a Notification Channel [REST_NetAPI_NotificationChannel].
4. The application stops the notifications subscription using DELETE with a resource URL containing the subscriptionId

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 Appendix H of [REST_NetAPI_ACR].
    - The ACR 'auth' 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: Inbound messages for a given registration

The resource used is:
http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages

This resource is for polling incoming messages using retrieval criteria that are set up in advance during provisioning process for a particular application.

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 client wants to use. The value of this variable is defined in section 5.1.
registrationId	Reference to the retrieval criteria provisioned in advance and known to the client application

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 RESTful Messaging API, see section 7.

6.1.3 GET

This operation is used for reliable inbound message retrieval for the particular client. Messages will remain on the server until client will confirm successful retrieval by executing DELETE method (section 6.4.6).

Note that if the retrieved inbound message includes a request for "Displayed" status report (element "reportRequest" is present), the receiver application SHALL use the method described in 6.15.4 to inform the server when the message has been displayed.

Supported parameters in the query string of the request URL parameters are:

Name	Type/Values	Optional	Description
maxBatchSize	xsd:int	Yes	Specifies maximum number of messages to be returned in the response.
retrievalOrder	RetrievalOrder	Yes	Specifies order in which messages SHOULD be retrieved if there are more then one pending.
useAttachmentURLs	xsd:boolean	Yes	Default: false. If set to 'true', inbound message would have links to attachments together with the indication of the content type and optionally the size of each attachment. Otherwise, only message identifier will be returned, so that individual message retrieval can be done.
priority	MessagePriority	Yes	The priority of the messages to poll from the gateway. All messages of the specified priority and higher will be retrieved. If not specified, all messages shall be returned, i.e. the same as specifying Low.

6.1.3.1 Examples 1: Retrieve messages for a registration, useAttachmentURLs=false (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <!-- MMS  -->
 <inboundMessage>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <senderAddress>tel:+19585550101</senderAddress>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
  <messageId>msg123</messageId>
  <inboundMMSMessage>
   <subject>Who is RESTing on the beach?</subject>
  </inboundMMSMessage>
 </inboundMessage>
```



```
 <!-- MMS  -->
 <inboundMessage>
   <destinationAddress>tel:+19585550102</destinationAddress>
   <senderAddress>tel:+19585550103</senderAddress>
   <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg124</resourceURL>
   <messageId>msg124</messageId>
   <inboundMMSMessage>
     <subject>Who is RESTing on the beach?</subject>
   </inboundMMSMessage>
 </inboundMessage>
 <totalNumberOfPendingMessages>20</totalNumberOfPendingMessages>
 <numberOfMessagesInThisBatch>2</numberOfMessagesInThisBatch>

<resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2</resourceURL></
msg:inboundMessageList>
```

6.1.3.2 Example 2: request with invalid (non-existing) id (Informative)

6.1.3.2.1 Request

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.2.2 Response

```
HTTP/1.1 404 Not Found
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
 <link rel="self"   href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2" />
 <serviceException>
   <messageId>SVC0004</messageId>
   <text>No valid addresses provided in message part %1</text>
   <variables>reg123</variables>
 </serviceException>
</common:requestError>
```

6.1.3.3 Example 3: Retrieve messages with attachment URLs (Informative)

6.1.3.3.1 Request

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2&useAttachmentURLs=true HTTP/1.1
Accept: application/xml
Host: example.com
```


6.1.3.3.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <!-- MMS  -->
 <inboundMessage>
   <destinationAddress>tel:+19585550100</destinationAddress>
   <senderAddress>tel:+19585550101</senderAddress>
   <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
   <messageId>msg123</messageId>
   <inboundMMSMessage>
     <subject>Who is RESTing on the beach?</subject>
     <attachment>
       <contentType>image/gif</contentType>
       <size>27000</size>
       <link rel="attachment"
        href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123
        /attachments/attach123" />

     </attachment>
     <attachment>
       <contentType>video/3gpp</contentType>
       <size>125000</size>
       <link rel="attachment"
        href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123
        /attachments/attach124" />
</attachment>
     <bodyText>See attached picture</bodyText>
   </inboundMMSMessage>
 </inboundMessage>
 <totalNumberOfPendingMessages>20</totalNumberOfPendingMessages>
 <numberOfMessagesInThisBatch>2</numberOfMessagesInThisBatch>
 <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/
msg123/attachments</resourceURL>
</msg:inboundMessageList>
```

6.1.3.4 Example 4: maxBatchSize exceeding the allowed size (Informative)

6.1.3.4.1 Request

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=5000 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.4.2 Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="InboundMessageList"
    href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=5000" />
  <policyException>
    <messageId>POL1020</messageId>
    <text>MaxBatchSize exceeded. The maximum allowed maxBatchSize is %1.</text>
    <variables>20</variables>
  </policyException>
</common:requestError>
```

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.1.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.2 Resource: Inbound messages retrieve and delete using registration

The resource used is:
http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages

This resource is used for retrieving and deleting the list of incoming messages using retrieval criteria that are set up in advance (offline - during provisioning process: short codes, etc) for a particular client.

After this step, attachments or individual messages are still available for the individual retrieval.

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 client wants to use. The value of this variable is defined in section 5.1.
registrationId	Reference to the retrieval criteria provisioned in advance and known to the client application

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 RESTful Messaging API, see section 7.

6.2.3 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.2.4 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.2.5 POST

This operation retrieves one or more messages from the gateway storage for the particular client. If retrieval is successful, it will delete message from gateway.

Notes: POST is used because resource state would be altered as result of the execution. GET is not a good fit here because it has to be idempotent. Client guidelines:

1) should NOT be used for reliable message delivery (see GET for reliable delivery). This is an optional alternative to the use of GET and DELETE on the …/inbound/subscriptions resource.

2) Default number of messages that would be returned in one batch is controlled by server configuration.

3) Messages would be deleted from gateway storage following a successful POST, after a maximum time interval as defined by a service policy. Client needs to retrieve body of the message with all attachments by executing separate POST using URLs provided in 'link' in response, if useAttachmentURLs was set to 'true' in the request.

Parameters are passed in the request body using the InboundMessageRetrieveAndDeleteRequest data structure.

Note that if the retrieved inbound message includes a request for “Displayed” status report (element “reportRequest” is present), the receiver application SHALL use the method described in 6.15.4 to inform the server when the message has been displayed.

6.2.5.1 Example: Retrieve and delete inbound messages (Informative)

6.2.5.1.1 Request

```
POST /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/retrieveAndDeleteMessages HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageRetrieveAndDeleteRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <retrievalOrder>OldestFirst</retrievalOrder>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:inboundMessageRetrieveAndDeleteRequest>
```

6.2.5.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <!-- MMS  -->
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <!--  resourceURL is not included because message is deleted from the server already  -->
    <messageId>msg123</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
  <!-- MMS  -->
  <inboundMessage>
    <destinationAddress>tel:+19585550102</destinationAddress>
    <senderAddress>tel:+19585550103</senderAddress>
    <!--  resourceURL is not included because message is deleted from the server already -->
    <messageId>msg124</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
  <totalNumberOfPendingMessages>20</totalNumberOfPendingMessages>
  <numberOfMessagesInThisBatch>2</numberOfMessagesInThisBatch>
<resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/retrieveAndDeleteMessages</resourceURL>
</msg:inboundMessageList>
```

6.2.6 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.3 Resource: Retrieval and deletion of individual inbound message using registration

The resource used is:
http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}/retrieveAndDelete

This resource is used to retrieve and simultaneously delete individual inbound message and all attachments stored by the gateway, in MIME representation. It is an alternative way to get access to the message. GET followed by delete on **http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}** resource should be used for reliable delivery.

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 client wants to use. The value of this variable is defined in section 5.1.
registrationId	Reference to the retrieval criteria provisioned in advance and known to the client application
messageId	Unique message identifier generated by server

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 RESTful Messaging 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 sections 6.5.5 and 7.4.1 of [RFC7231].

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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.3.5 POST

This operation is used to read and delete one message from gateway storage. If successful, message would be deleted together with all associated attachments, after an agreed time interval as defined by a service policy.

Note that if the retrieved inbound message includes a request for "Displayed" status report (element "reportRequest" is present), the receiver application SHALL use the method described in 6.15.4 to inform the server when the message has been displayed.

6.3.5.1 Example: Read and delete one message (informative)

6.3.5.1.1 Request

```
POST /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/retrieveAndDelete HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageRetrieveAndDeleteRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:inboundMessageRetrieveAndDeleteRequest>
```

6.3.5.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Length: nnnn
Content-Type: multipart/form-data;,          boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name="root-fields"
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessage xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <destinationAddress>tel:+19585550100</destinationAddress>
 <senderAddress>tel:+19585550101</senderAddress>
 <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
 <messageId>msg123</messageId>
 <inboundMMSMessage>
  <subject>Who is RESTing on the beach?</subject>
 </inboundMMSMessage>
</msg:inboundMessage>

--===============123456==
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment;filename="textBody.txt";
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit

Look at the attached picture

--====aaabbb
```

```
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>

GIF89a...binary image data...

--====aaabbb--
--===============123456==--
```

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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.4 Resource: Inbound message for a given registration

The resource used is:
http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}

This resource provides access to individual inbound message stored by the gateway. Combination of GET/DELETE is used by clients that are polling incoming messages and require reliable delivery. Each message would have to be deleted separately as a confirmation of successful retrieval.

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 client wants to use. The value of this variable is defined in section 5.1.
registrationId	Reference to the provisioned in advance and known to the client application
messageId	Unique message identifier generated by server

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 RESTful Messaging API, see section 7.

6.4.3 GET

This operation is used to read one message from gateway storage. Message is not deleted. DELETE method needs to be executed to confirm delivery and free resources occupied by the message and associated attachments.

Note that if the retrieved inbound message includes a request for "Displayed" status report (element "reportRequest" is present), the receiver application SHALL use the method described in 6.15.4 to inform the server when the message has been displayed.

6.4.3.1 Example 1: Read message from gateway storage (Informative)

6.4.3.1.1 Request

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].

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123?resFormat=XML HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type:  multipart/form-data; boundary="=====12345===="
Content-Length: nnnn


--====12345====
Content-Disposition=multipart/form-data; name="root-fields"
Content-Type=application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessage xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <destinationAddress>tel:+19585550100</destinationAddress>
  <senderAddress>tel:+19585550101</senderAddress>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
  <messageId>msg123</messageId>
  <inboundMMSMessage>
    <subject>Who is RESTing on the beach?</subject>
  </inboundMMSMessage>
</msg:inboundMessage>

--====12345====
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment;filename="textBody.txt";
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit


Look at the attached picture


--====aaabbb
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>
```

```
GIF89a...binary image data...

--====aaabbb--
--===============123456==--
```

6.4.3.2 Example 2: Read message from gateway storage, Displayed status report requested (Informative)

6.4.3.2.1 Request

This example shows retrieval of an indound message which includes a request for “Displayed” status report.

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.3.2.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type:  multipart/form-data; boundary=”=====12345====”
Content-Length: nnnn


--====12345====
Content-Disposition=multipart/form-data; name=”root-fields”
Content-Type=application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessage xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <destinationAddress>tel:+19585550100</destinationAddress>
  <senderAddress>tel:+19585550101</senderAddress>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
  <link rel="MessageStatusReport"
href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status"/>
  <messageId>msg123</messageId>
  <reportRequest>Displayed</reportRequest >
  <inboundMMSMessage>
    <subject>Who is RESTing on the beach?</subject>
  </inboundMMSMessage>
</msg:inboundMessage>

--====12345====
Content-Disposition: form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”====aaabbb”
--====aaabbb
Content-Disposition:attachment;filename=”textBody.txt”;
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit
```



```
Look at the attached picture

--====aaabbb
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>

GIF89a...binary image data...

--====aaabbb--
--===============123456==--
```

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 sections 6.5.5 and 7.4.1 of [RFC7231].

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 sections 6.5.5 and 7.4.1 of [RFC7231].

Note: See Inbound MMS message retrieve and delete.

6.4.6 DELETE

Confirms message delivery and removes the message from the cache/storage on the gateway.

6.4.6.1 Example: Remove message from gateway storage (Informative)

6.4.6.1.1 Request

```
DELETE /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.6.1.2 Response

```
HTTP/1.1 204 No content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.5 Resource: Inbound message attachment

The resource used is:

http://{serverRoot}/messaging/{apiVersion}/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId}

This resource is used to provide access to individual MMS attachment stored by the gateway. Combination of GET/DELETE is used by clients that are polling incoming messages and require reliable delivery. Each attachment would have to be deleted separately as a confirmation of successful retrieval.

Individual deletions over all attachments would have the same effect as a DELETE over an individual message

(/inbound/registrations/{registrationId}/messages/{messageId}).

POST on ../retrieveAndDelete resource is used to pop (read and delete in the single step) MMS message (body+attachments) from the gateway storage. It would require no subsequent DELETE operations.

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 client wants to use. The value of this variable is defined in section 5.1.
registrationId	Reference to the retrieval criteria provisioned in advance and known to the client application
messageId	Unique message identifier generated by server
attachmentId	Unique attachment identifier generated by server

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 RESTful Messaging API, see section 7.

6.5.3 GET

This operation is used to Read one MMS attachment from the gateway storage. Attachment is not deleted. DELETE method need to be executed to confirm delivery and free resources occupied by the attachment.

6.5.3.1 Example: Read an MMS attachment (Informative)

6.5.3.1.1 Request

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: image/gif, image/png, image/jpeg, text/html, application/xml
Host: example.com
```

6.5.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Length: nnnn
Content-Type: image/gif

...GIF89a...binary image data
```

6.5.4 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.5.5 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.5.6 DELETE

This operation is used to confirm successful attachment retrieval and to remove it from the gateway storage.

6.5.6.1 Example: Delete an MMS attachment from gateway storage (Informative)

6.5.6.1.1 Request

```
DELETE /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.5.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.6 Resource: Inbound message subscriptions

The resource used is: **http://{serverRoot}/messaging/{apiVersion}/inbound/subscriptions**

This resource gives access to inbound subscriptions for a particular client.

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.

6.6.1 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 client wants to use. The value of this variable is defined in section 5.1.

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 RESTful Messaging API, see section 7.

6.6.3 GET

This operation is used to read active subscriptions for the particular client.

6.6.3.1 Example: Read active subscriptions (Informative)

6.6.3.1.1 Request

```
GET /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
```

6.6.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscriptionList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <subscription>
  <callbackReference>
   <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/12345</notifyURL>
   <callbackData>12345</callbackData>
  </callbackReference>
  <destinationAddress>tel:+19585550101</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567891</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/0000001</resourceURL>
  <useAttachmentURLs>false</useAttachmentURLs>
 </subscription>
 <subscription>
  <callbackReference>
   <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/54321</notifyURL>
   <callbackData>54321</callbackData>
   <notificationFormat>XML</notificationFormat>
  </callbackReference>
  <destinationAddress>tel:+19585550102</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567892</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/0000002</resourceURL>
  <useAttachmentURLs>false</useAttachmentURLs>
 </subscription>
 <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions</resourceURL>
</msg:subscriptionList>
```

6.6.4 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.6.5 POST

This operation is used to create a new inbound message subscription for the particular client.

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]).

6.6.5.1 Example 1: Create inbound subscription, returning a representation of created resource (Informative)

6.6.5.1.1 Request

```
POST /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/88888</notifyURL>
    <callbackData>12345</callbackData>
  </callbackReference>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567893</clientCorrelator>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:subscription>
```

6.6.5.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/88888</notifyURL>
    <callbackData>12345</callbackData>
  </callbackReference>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567893</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123</resourceURL>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:subscription>
```

6.6.5.2 Example 2: Create inbound subscription, returning the location of created resource (Informative)

6.6.5.2.1 Request

```
POST /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/88888</notifyURL>
    <callbackData>12345</callbackData>
  </callbackReference>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <criteria>Urgent*</criteria>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:subscription>
```

6.6.5.2.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123</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: GET, POST’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.7 Resource: Individual inbound message subscription

The resource used is: **http://{serverRoot}/messaging/{apiVersion}/inbound/subscriptions/{subscriptionId}**

This resource controls individual subscription for inbound messages for a particular client.

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 client wants to use. The value of this variable is defined in section 5.1.
subscriptionId	Identifies the subscription

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 RESTful Messaging API, see section 7.

6.7.3 GET

This operation is used to read an individual subscription for the particular client.

6.7.3.1 Example: Read individual subscription (Informative)

6.7.3.1.1 Request

```
GET /exampleAPI/messaging/v1/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.7.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/88888</notifyURL>
    <notificationFormat>XML</notificationFormat>
  </callbackReference>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567893</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123</resourceURL>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:subscription>
```

6.7.4 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.7.5 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.7.6 DELETE

This operation is used to delete a subscription for the particular client.

6.7.6.1 Example: Delete a subscription (Informative)

6.7.6.1.1 Request

```
DELETE /exampleAPI/messaging/v1/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.7.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.8 Resource: Client notification about inbound message

This resource is a callback URL provided by the client for notification about incoming messages. The RESTful Messaging 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 to obtain the notifications, the client needs to use the mechanisms described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.8.5.

6.8.1 Request URL variables

Client provided.

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 RESTful Messaging API, see section 7.

6.8.3 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.8.4 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.8.5 POST

This operation is used to notify client about message arrival. Note that if the message includes request for "Displayed" message status report, the receiver application SHALL use the method described in 6.15.4 to inform the server when the message has been displayed.

6.8.5.1 Example 1: Message arrival notification (Informative)

6.8.5.1.1 Request

```
POST /notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/xml
Host: application.example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageNotification xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
    <link rel="Subscription"  href="http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123"/>
    <messageId>msg123</messageId>
    <inboundMMSMessage>
       <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
</msg:inboundMessageNotification>
```

6.8.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.8.5.2 Example 2: Message arrival notification with attachment URLs (Informative)

6.8.5.2.1 Request

```
POST /notifications/DeliveryInfoNotification/8888 HTTP/1.1
Accept: application/xml
Host: application.example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageNotification xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
    <link rel="Subscription" href="http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123" />
    <messageId>msg123</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
```



```
      <attachment>
         <contentType>image/gif</contentType>
         <size>27000</size>
         <link rel="attachment"
          href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123
          /attachments/attach123" />
      </attachment>
      <attachment>
         <contentType>video/3gpp</contentType>
         <size>125000</size>
         <link rel="attachment"
          href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123
          /attachments/attach124" />
      </attachment>
      <bodyText>Look at the attached picture</bodyText>
   </inboundMMSMessage>
  </inboundMessage>
</msg:inboundMessageNotification>
```

6.8.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.8.5.3 Example 3: Message (with Displayed status report requested) arrival notification (Informative)

6.8.5.3.1 Request

```
POST /notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/xml
Host: application.example.com
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:inboundMessageNotification xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123</resourceURL>
    <link rel="MessageStatusReport"
href="http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status"/>
    <link rel="Subscription"  href="http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123"/>
    <messageId>msg123</messageId>
    <reportRequest>Displayed</reportRequest>
    <inboundMMSMessage>
       <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
</msg:inboundMessageNotification>
```

6.8.5.3.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.8.6 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.9 Resource: Outbound message requests

The resource used is: **http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests**

This resource is used for sending outbound messages.

In the case an optional notification URL is passed to the server when creating an outbound message request, 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 the outbound request.

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 client wants to use. The value of this variable is defined in section 5.1.
senderAddress	Sender identifier. Examples: 72654 (SHORT CODE [REST_NetAPI_Common]), 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 RESTful Messaging API, see section 7.

6.9.3 GET

This operation is used to retrieve the list of pending outgoing requests.

6.9.3.1 Example: Retrieve list of outgoing requests (Informative)

6.9.3.1.1 Request

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


6.9.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequestList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <outboundMessageRequest>
  <address>tel:+19585550103</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <outboundMMSMessage>
     <subject>Holiday greetings</subject>
  </outboundMMSMessage>
  <clientCorrelator>567894</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
  <deliveryInfoList>

<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos</resour
ceURL>
    <deliveryInfo>
     <address>tel:+19585550103</address>
     <deliveryStatus>DeliveredToTerminal</deliveryStatus>
    </deliveryInfo>
  </deliveryInfoList>
  </outboundMessageRequest>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests</resourceURL>
</msg:outboundMessageRequestList>
```

6.9.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405.The returned HTTP error status is 405. The server should also include the ‘Allow: GET, POST’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.9.5 POST

This operation is used to create outgoing message request. It must follow the serialization guidelines described in section 5.6 of [REST_WP] in order to combine the multiple MIME body parts into the HTTP request message.

The notifyURL in the optional receiptRequest 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]).

6.9.5.1 Example 1: Create outgoing message, returning the representation of created resource (Informative)

6.9.5.1.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</msg:outboundMessageRequest>
```

6.9.5.2 Example 2: Create outgoing message, returning the location of created resource (Informative)

6.9.5.2.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
```

```
   <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.2.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL> http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</common:resourceReference>
```

6.9.5.3 Example 3: Create outgoing message with charging (Informative)

6.9.5.3.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/xml
Content-Length : nnnn

<?xml version="1.0" encoding="UTF-8"?>
```

```
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <charging>
    <description>Sample text for the charging information</description>
  </charging>
  <receiptRequest><!-- this is optional -->
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567896</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==
Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="text.txt"
Content-Type: text/plain
Content-Length: nnnn

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif
Content-Length: nnnn

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.3.2 Response for charging not supported

```
HTTP/1.1 400 Bad request
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <policyException>
    <messageId>POL0008</messageId>
    <text>Charging is not supported</text>
  </policyException>
</common:requestError>
```

6.9.5.4 Example 4: Create outgoing message, serviceException in case of address(es) failure (Informative)

6.9.5.4.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.4.2 Response

```
HTTP/1.1 400 Bad Request
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <serviceException>
    <messageId>SVC0004</messageId>
    <text>No valid addresses provided in message part %1</text>
    <variables>variables</variables>
  </serviceException>
</common:requestError>
```

6.9.5.5 Example 5: Create outgoing message, multiple addresses partial success, with deliveryInfoList in response (Informative)

6.9.5.5.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==
Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
```

```
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.5.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
      <subject> hello from the rest of us!</subject>
      <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
  <deliveryInfoList>

<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos</resour
ceURL>
    <deliveryInfo>
      <address>tel:+19585550103</address>
      <deliveryStatus>MessageWaiting</deliveryStatus>
    </deliveryInfo>
    <deliveryInfo>
      <address>tel:+19585550104</address>
      <deliveryStatus>DeliveryImpossible</deliveryStatus>
    </deliveryInfo>
  </deliveryInfoList>
</msg:outboundMessageRequest>
```


6.9.5.6 Example 6: Create outgoing message, multiple addresses partial success, without deliveryInfoList in response (Informative)

6.9.5.6.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.6.2 Response

Note: In this case, in order to know the result of sending to individual addresses, the delivery status can be obtained using the GET operation with the requestId, or via notifications (if subscribed).

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress> tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
      <subject> hello from the rest of us!</subject>
      <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</msg:outboundMessageRequest>
```

6.9.5.7 Example 7: using SHORT CODE as senderAddress (Informative)

6.9.5.7.1 Request

```
POST /exampleAPI/messaging/v1/outbound/72654/requests HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
Content-Length: nnnn
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>72654</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
```



```
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</msg:outboundMessageRequest>

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

6.9.5.7.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/72654/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>72654</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/72654/requests/req123</resourceURL>
</msg:outboundMessageRequest>
```


6.9.6 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.10 Resource: Outbound message request and delivery status

The resource used is: **http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests/{requestId}**

This resource is used to retrieve an outbound message request including the message delivery status.

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 client wants to use. The value of this variable is defined in section 5.1.
senderAddress	Sender identifier. Examples: 72654 (SHORT CODE [REST_NetAPI_Common]), tel:+19585550100, acr:pseudonym123
requestId	Outbound message request identifier generated by server

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 RESTful Messaging API, see section 7.

6.10.3 GET

This operation is used to retrieve an outbound message request including the message delivery status.

6.10.3.1 Example: Read message request and delivery status (Informative)

6.10.3.1.1 Request

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.10.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <!-- this is optional -->
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>Holiday greetings</subject>
 </outboundMMSMessage>
 <clientCorrelator>567895</clientCorrelator>
 <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
  <deliveryInfoList>
     <!-- this is optional -->

<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfos</resourceURL>
     <deliveryInfo>
      <address>tel:+19585550103</address>
      <deliveryStatus>MessageWaiting</deliveryStatus>
     </deliveryInfo>
     <deliveryInfo>
      <address>tel:+19585550104</address>
      <deliveryStatus>MessageWaiting</deliveryStatus>
     </deliveryInfo>
  </deliveryInfoList>
</msg:outboundMessageRequest>
```

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.10.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.11 Resource: Outbound message delivery status

The resource used is:

http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/requests/{requestId}/deliveryInfos

This resource is used to request outbound message delivery status.

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 client wants to use. The value of this variable is defined in section 5.1.
senderAddress	Sender identifier. Examples: 72654 (SHORT CODE [REST_NetAPI_Common]), tel:+19585550100, acr:pseudonym123
requestId	Outbound message request identifier generated by server

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 RESTful Messaging API, see section 7.

6.11.3 GET

This operation is used to retrieve outgoing message delivery status.

6.11.3.1 Example: Read message delivery status (Informative)

6.11.3.1.1 Request

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos HTTP/1.1
Accept: application/xml
Host: example.com
```

6.11.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryInfoList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">

<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos</resourceURL>
   <deliveryInfo>
     <address>tel:+19585550103</address>
     <deliveryStatus>MessageWaiting</deliveryStatus>
   </deliveryInfo>
   <deliveryInfo>
     <address>tel:+19585550104</address>
     <deliveryStatus>MessageWaiting</deliveryStatus>
   </deliveryInfo>
</msg:deliveryInfoList>
```

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.11.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

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’ field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.12 Resource: Outbound message delivery notification subscriptions

The resource used is: **http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/subscriptions**

This resource gives access to outbound subscriptions for a particular client.

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.

6.12.1 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 client wants to use. The value of this variable is defined in section 5.1.
senderAddress	Sender identifier. Examples: 72654 ([REST_NetAPI_Common]), tel:+19585550100, acr:pseudonym123

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 RESTful Messaging API, see section 7.

6.12.3 GET

This operation is used to read all outbound message delivery notification subscriptions for the particular client.

6.12.3.1 Example: Read delivery notification subscriptions (Informative)

6.12.3.1.1 Request

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

6.12.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscriptionList xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
 <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions</resourceURL>
 <deliveryReceiptSubscription>
  <callbackReference>
   <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
   <callbackData>12345</callbackData>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123</resourceURL>
 </deliveryReceiptSubscription>
 <deliveryReceiptSubscription>
  <callbackReference>
   <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77778</notifyURL>
   <callbackData>54321</callbackData>
  </callbackReference>
  <filterCriteria>0103</filterCriteria>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123</resourceURL>
 </deliveryReceiptSubscription>
</msg:deliveryReceiptSubscriptionList>
```

6.12.4 PUT

Method not supported 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.12.5 POST

This operation is used to create a new outbound message delivery notification subscription for the particular client.

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]).

6.12.5.1 Example 1: Create outbound delivery notification subscription using 'tel' URI (Informative)

6.12.5.1.1 Request

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

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
</msg:deliveryReceiptSubscription>
```

6.12.5.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123</resourceURL>
</msg:deliveryReceiptSubscription>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.6.5.2.2).

6.12.5.2 Example 2: Create outbound delivery notification subscription using 'acr' URI (Informative)

6.12.5.2.1 Request

```
POST /exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
</msg:deliveryReceiptSubscription>
```

6.12.5.2.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123</resourceURL>
</msg:deliveryReceiptSubscription>
```

6.12.6 DELETE

Method not supported 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.13 Resource: Individual outbound message delivery notification subscription

The resource used is:

http://{serverRoot}/messaging/{apiVersion}/outbound/{senderAddress}/subscriptions/{subscriptionId}

This resource controls individual subscription for outbound message delivery notification and gives access to individual subscription for a particular client.

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 client wants to use. The value of this variable is defined in section 5.1.
senderAddress	Sender identifier. Examples: 72654 (SHORT CODE [REST_NetAPI_Common]), tel:+19585550100, acr:pseudonym123
subscriptionId	Identifier of the subscription

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

6.13.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

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

6.13.3 GET

This operation is used to read an individual outbound message delivery notification subscription for the particular client.

6.13.3.1 Example: Read individual message delivery notification subscription (Informative)

6.13.3.1.1 Request

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.13.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123</resourceURL>
</msg:deliveryReceiptSubscription>
```

6.13.4 PUT

Method not supported 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.13.5 POST

Method not supported 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.13.6 DELETE

This operation is used to delete a subscription for the particular client.

6.13.6.1 Example: Delete message delivery notification subscription (Informative)

6.13.6.1.1 Request

```
DELETE /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.13.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.14 Resource: Client notification about outbound message delivery status

This resource is a callback URL provided by the client for notification about outbound message delivery status. The RESTful Messaging 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 to obtain the notifications, the client needs to use the mechanisms described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.14.5.

To outbound message delivery status notifications the following table applies:

EventType	Notification Root Element Type	Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/messaging/{apiVersion}
n/a	DeliveryInfoNotification	Sender of the message	n/a	OutboundMessageRequest	/outbound/{senderAddress}/requests/{requestId}

The resource URL of the resource representing the underlying outbound message request is passed in the “href” attribute of the “link” element with rel=”OutboundMessageRequest”.

6.14.1 Request URL variables

Client provided.

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 RESTful Messaging API, 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 sections 6.5.5 and 7.4.1 of [RFC7231].

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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.14.5 POST

This operation is used to notify the client about outbound message delivery status

6.14.5.1 Example 1: Notify client about outbound message delivery status, multiple delivery status per notification (Informative)

6.14.5.1.1 Request

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

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryInfoNotification xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <deliveryInfo>
    <address>tel:+19585550103</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
  </deliveryInfo>
  <deliveryInfo>
    <address>tel:+19585550104</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
</deliveryInfo>
  <link rel="OutboundMessageRequest"
  href="http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123"/>
</msg:deliveryInfoNotification>
```

6.14.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.14.5.2 Example 2: Notify client about outbound message delivery status, single delivery status per notification (Informative)

6.14.5.2.1 Request

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

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryInfoNotification xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <deliveryInfo>
    <address>tel:+19585550104</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
</deliveryInfo>
  <link rel="OutboundMessageRequest"
    href="http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123"/>
</msg:deliveryInfoNotification>
```


6.14.5.2.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.15 Resource: Individual inbound message status

The resource URL is provided by the server and it is included in the "href" attribute of the "link" element with rel="MessageStatusReport" in the received "InboundMessage". The RESTful Messaging API does not make any assumption about the structure of this URL.

This resource represents the status of a message.

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

6.15.1 Request URL variables

Server 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 RESTful Messaging API, 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: PUT' field in the response as per sections 6.5.5 and 7.4.1 of [RFC7231].

6.15.4 PUT

This operation is used for reporting the "Displayed" status of a message. The resource URL of the resource to be updated is provided by the server as described above. 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 "DeliveredToTerminal" status report is generated by the API Server by procedures of the underlying protocol layers which are out of scope of this specification.

6.15.4.1 Example: Reporting the status of an inbound message (Informative)

6.15.4.1.1 Request

```
PUT /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

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


6.15.4.1.2 Response

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

6.15.5 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 sections 6.5.5 and 7.4.1 of [RFC7231].

6.15.6 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 sections 6.5.5 and 7.4.1 of [RFC7231].

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common]. The following additional Service Exception codes are defined for the RESTful Messaging API.

7.1.1 SVC0283: Delivery Receipt Notification not supported

Name	Description
MessageID	SVC0283
Text	Delivery Receipt Notification not supported
Variables	None
HTTP status code(s)	403 Forbidden

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common].

The following additional Policy Exception codes are defined for the RESTful Short Messaging API.

7.2.1 POL1019: Binary SMS not allowed

Name	Description
MessageID	POL1019
Text	Binary SMS is not allowed.
Variables	None
HTTP status code(s)	403 Forbidden

7.2.2 POL1020: MaxBatchSize exceeded

Name	Description
MessageID	POL1020
Text	MaxBatchSize exceeded. The maximum allowed maxBatchSize is %1.
Variables	%1 Allowed maximum value for maxBatchSize
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 Versions REST_NetAPI_Messaging-V1_0	28 Apr 2011	Many	Structural changes to fit the OMA RESTful Network API release. This version inherits the technical content of OMA-TS-ParlayREST_MultiMediaMessaging-V1_1-20110111-C and applies changes according to ARC INP 30R01, 98R02, 155R01,156R0, 0186, 0187R02, and 164R04
	10 May 2011	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2011-007-CR_TS_changes_for_Messaging, OMA-ARC-2011-0208-CR_Messaging_JSON_statement
	27 May 2011	Many	Implemented CR: OMA-ARC-REST-NetAPI-2011-0027R04-CR_Messaging_TS_Evolution_to_agreed_new_template
	06 Jun 2011	Many	Implemented AI: REST-NetAPI-2011-A025
	22 Jun 2011	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2011-0052-CR_TS_Messaging_Replace_MMS_by_MSG_in_SCRs and OMA-ARC-REST-NetAPI-2011-0058-CR_Messaging_TS_fix_encoding_rules_and_examples
	28 Jun 2011	Many	Implemented CR: OMA-ARC-REST-NetAPI-2011-0079R02-CR_Messaging_ContentType_in_AttachmentURL
	01 Jul 2011	Contents	Sync'ed ToC with document changes
	23 Jul 2011	Many	Implemented CR: OMA-ARC-REST-NetAPI-2011-0125-CR_Messaging_examples_bug_fixes
	28 Jul 2011	Many	Implemented CR: OMA-ARC-REST-NetAPI-2011-0159R01-CR_Messaging_telURI_fixes, OMA-ARC-REST-NetAPI-2011-0162R01-CR_Messaging_Notif_Channel_changes
	03 Aug 2011	Many	Implemented CR OMA-ARC-REST-NetAPI-2011-0176-CR_Messaging_Appendix_C_link_to_section_6 and REST-NetAPI-2011-A090 Added hyperlinks in SCR tables. Fixed editorials.
	24 Aug 2011	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0192-CR_Messaging_resourceURL
	01 Sep 2011	Many	Implemented CR OMA-ARC-REST-NetAPI-2011-0199R04-CR_Messaging_ACR
	04 Oct	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0267-CR_Messaging_prep_for_CONR
	18 Oct 2011	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0257-CR_Bkw_compatibility_Messaging_TS, OMA-ARC-REST-NetAPI-2011-0295R01-CR_Messaging_bug_fixes_before_CONR
	09 Nov 2011	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0334-CR_Messaging_implement_CR_326_versioning_blueprint, OMA-ARC-REST-NetAPI-2011-0354-CR_Messaging_CR325_resourceURL_blueprint
	28 Nov 2011	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0382R02-CR_Messaging_Appendix_G_and_global_comment
	12 Dec 2011	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0438R01-CR_Messaging_acr_Autho_and_misc_CONR
	10 Jan 2012	Many	Implemented: OMA-ARC-REST-NetAPI-2011-0463-CR_Messaging_CONR_comments_resolutions
	12 Jan 2012	All	Changed document from file type ".docx" to ".doc". Changed copyright dates to 2012. Editorial changes
	18 Jan 2012	5.1	Editorial, corrected labels in resource hierarchy diagram. Restored n cover page the hyperlinks to the font/color as per Template
Candidate Version	31 Jan 2012	n/a	Status changed to Candidate by TP

Document Identifier	Date	Sections	Description
REST_NetAPI_Messaging-V1_0			TP Ref # OMA-TP-2012-0018-INP_REST_NetAPI_Messaging_1_0_ERP_and_ETR_for_Candidate_Approval
Draft Versions REST_NetAPI_Messaging-V1_0	20 Jul 2012	5, 5.3, 5.3.2, 6.1.1, 6.1.2, 6.2.1, 6.2.2, 6.3.1, 6.3.2, 6.4.1, 6.4.2, 6.5.1, 6.5.2, 6.6.1, 6.6.2, 6.7.1, 6.7.2, 6.8.2, 6.9.1, 6.9.2, 6.10.1, 6.10.2, 6.11.1, 6.11.2, 6.12.1, 6.12.2, 6.13.1, 6.13.2, 6.14.2, 7, C.1, D.15, D.16, G.1, G.1.1.1, G.1.1.3	Incorporated CRs: OMA-ARC-REST-NetAPI-2012-0100R01-CR_Fix_senderAddress_and_close_A008_for_Messaging OMA-ARC-REST-NetAPI-2012-0136-CR_Messaging_POST_or_NOTIFY OMA-ARC-REST-NetAPI-2012-0169-CR_Messaging_TS_Adding_Section7_A054 Editorial changes
	22 Aug 2012	5.2.2.8, 5.2.2.11, 5.2.2.23, C.1, C.2, C.3	Incorporated CR: OMA-ARC-REST-NetAPI-2012-0223R01-CR_Messaging_clientCorrelator_resolution
	02 Oct 2012	5.2.2.11, 5.2.2.18, 6.1.3.2.2, 6.1.3.4, 6.9.5.4.2, 7.2, D.2, D.4, D.22	Incorporated CRs: OMA-ARC-REST-NetAPI-2012-0213-CR_Followup_for_INP_200_TS_Messaging OMA-ARC-REST-NetAPI-2012-0264-CR_Messaging_support_for_Flash_SMS Editorial changes
	22 Oct 2012	5.1, 6.1.3.1.2, 6.1.3.2.2, 6.1.3.3.2, 6.1.3.4.2, 6.2.5.1.2, 6.8.5.2.1, 6.9.3.1.2, 6.9.5.5.2, 6.11.3.1.2, 6.14.5.1.1, D.1, D.3, D.5, D.18, D.23	Incorporated CR: OMA-ARC-REST-NetAPI-2012-0267R01-CR_Fixing_editorials_in_resource_hierarchy_diagram_and_examples Template change to OMA-Template-Spec-RESTNetAPI-20120813-I Editorial changes
	17 Dec 2012	2.2, 3.2,	Incorporated CR:

Document Identifier	Date	Sections	Description
		4.1, 5, 5.3.2, 6, 6.8, 6.14, G.1.2,	OMA-ARC-REST-NetAPI-2012-0294-CR_Messaging_apply_blueprints_for_auth_and_longpolling
	09 Feb 2013	6.1.3.2.2, 6.1.3.3.2, 6.1.3.4.2, 6.8.5.2.1	Incorporated CR: OMA-ARC-REST-NetAPI-2013-0017-CR_Messaging_TS_validation_fix Template updated.
	26 Feb 2013	5.2.2.2, 5.2.2.7, 5.2.5	Incorporated CR: OMA-ARC-REST-NetAPI-2013-0018R03-CR_support_for_InboundVoicemail Editorial changes
	12 Mar 2013	4.1	Incorporated CR: OMA-ARC-REST-NetAPI-2013-0027-CR_add_VM_to_introduction Editorial changes
	20 Mar 2013	5.2.2.12, 5.2.2.21, 5.2.5	Incorporated CR: OMA-ARC-REST-NetAPI-2013-0023R02-CR_Messaging_adding_outbound_bearer_agnostic_message_type Editorial changes
Candidate Version REST_NetAPI_Messaging-V1_0	09 Jul 2013	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2013-0211-INP_REST_NetAPI_Messaging_V1_0_ERP_for_Candidate_re_approval
Draft Versions REST_NetAPI_Messaging-V1_0	13 Aug 2014	2.1, 5.2.2.2, 5.2.2.14	Incorporated CR: OMA-ARC-REST-NetAPI-2014-0058-CR_MMS_TS_adding_support_for_dateTimeStamp Editorial changes
	14 Oct 2014	2.1, 6	Incorporated CR: OMA-ARC-REST-NetAPI-2014-0073-CR_ACR_reference_in_TS_Messaging
	18 Dec 2014	4.1, 5.1, 5.2.2.1, 5.2.2.2, 5.2.2.3, 5.2.2.12, 5.2.2.21, 5.2.2.23, 5.2.2.24, 5.2.2.28, 5.2.3.1, 5.2.4, 5.3.1, 5.3.2, 5.3.3, 5.3.4, 6.1.3, 6.2.5, 6.3.5, 6.4.3, 6.4.3.1, 6.4.3.2, 6.8, 6.8.5, 6.8.5.3, 6.14, 6.15, B.1.14, B.1.15, D.8, D.19, D.36, D.37, G.1.1.3	Incorporated CR: OMA-ARC-REST-NetAPI-2014-0056R01-CR_MMS_TS_support_for_Displayed_message_status
	04 Mar 2015	2.1, 6.1.4, 6.1.5,	Incorporated CR: OMA-ARC-REST-NetAPI-2015-0023-CR_Messaging_TS_updating_references

Document Identifier	Date	Sections	Description
		6.1.6, 6.2.3, 6.2.4, 6.2.6, 6.3.3, 6.3.4, 6.3.6, 6.4.4, 6.4.5, 6.5.4, 6.5.5, 6.6.4, 6.6.6, 6.7.4, 6.7.5, 6.8.3, 6.8.4, 6.8.6, 6.9.4, 6.9.6, 6.10.4, 6.10.5, 6.10.6, 6.11.4, 6.11.5, 6.11.6, 6.12.4, 6.12.6, 6.13.4, 6.13.5, 6.14.3, 6.14.4, 6.14.6, 6.15.3, 6.15.5, 6.15.6, D, G.1.2	Editorial changes
	05 May 2015	4.1, 5.2.2.12, 5.2.2.21	Incorporated CR: OMA-ARC-REST-NetAPI-2015-0047R01-CR_outboundVM Editorial changes
Candidate Version REST_NetAPI_Messaging-V1_0	01 Dec 2015	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2015-0199-INP_REST_NetAPI_Messaging_V1_0_ERP_for_Candidate_re_approval

Appendix B. Static Conformance Requirements (Normative)

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

B.1 SCR for REST.MSG Server

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

B.1.1 SCR for REST.MSG.Inbound.Registration Server

Item	Function	Reference	Requirement
REST-MSG-INB-OFF-S-001-M	Support for reliable inbound messages delivery	6.1	
REST-MSG-INB-OFF-S-002-M	Retrieve messages from server - GET	6.1.3	

B.1.2 SCR for REST.MSG.Inbound.Registration.RetrieveDelete Server

Item	Function	Reference	Requirement
REST-MSG-INB-OFF-RETDEL-S-001-O	Support for inbound message delivery and delete	6.2	REST-MSG-INB-OFF-RETDEL-S-002-O
REST-MSG-INB-OFF- RETDEL-S-002-O	Retrieve and delete messages from server - POST	6.2.5	

B.1.3 SCR for REST.MSG.Individual.Inbound.Registration.RetrieveDelete Server

Item	Function	Reference	Requirement
REST-MSG-MIME-INB-OFF-RETDEL-S-001-O	Support for inbound message delivery and delete	6.3	REST-MSG-MIME-INB-OFF- RETDEL-S-002-O
REST-MSG-MIME-INB-OFF- RETDEL-S-002-O	Retrieve and delete one message from server - POST	6.3.5	

B.1.4 SCR for REST.MSG.Individual.Inbound Server

Item	Function	Reference	Requirement
REST-MSG-IND-INB-S-001-M	Support for inbound individual message delivery	6.4	
REST-MSG-IND-INB-S-002-O	Retrieve one message from server - GET	6.4.3	
REST-MSG-IND-INB-S-003-M	Confirm and delete retrieved message from server - DELETE	6.4.6	

B.1.5 SCR for REST.MSG.Attach.Individual.Inbound Server

Item	Function	Reference	Requirement
REST-MSG-ATTACH-IND-INB-S-001-O	Support for inbound individual message attachment delivery	6.5	REST-MSG-ATTACH-IND-INB-S-002-O AND REST-MSG-ATTACH-IND-INB-S-003-O
REST-MSG-ATTACH-IND-INB-S-002-O	Retrieve one message attachment from server - GET	6.5.3	
REST-MSG-ATTACH-IND-INB-S-003-O	Confirm and delete retrieved message attachment from server - DELETE	6.5.6	

B.1.6 SCR for REST.MSG.Inbound.Subscr Server

Item	Function	Reference	Requirement
REST-MSG-INB-ONL-SUBSCR-S-001-M	Support inbound subscriptions	6.6	
REST-MSG-INB-ONL-SUBSCR-S-002-O	Read active subscriptions - GET	6.6.3	
REST-MSG-INB-ONL-SUBSCR-S-003-M	Create inbound message subscription – POST (XML and JSON)	6.6.5	
REST-MSG-INB-ONL-SUBSCR-S-004-O	Create inbound message subscription – POST (application/x-www-form-urlencoded)	C.3	

B.1.7 SCR for REST.MSG.Inbound.Individual.Subscr Server

Item	Function	Reference	Requirement
REST-MSG-INB-INDON-SUBSCR-S-001-M	Support for control and read access to individual inbound subscription	6.7	
REST-MSG-INB-INDON- SUBSCR-S-002-O	Read individual inbound subscription - GET	6.7.3	
REST-MSG-INB-INDON- SUBSCR-S-003-M	Update individual inbound subscriptions - DELETE	6.7.6	

B.1.8 SCR for REST.MSG.Inbound.Notifications Server

Item	Function	Reference	Requirement
REST-MSG-INB-NOTIF-S-001-M	Support for notifying application about inbound messages	6.8	
REST-MSG-INB-NOTIF-S-002-M	Notify application about inbound message arrival - POST	6.8.5	

B.1.9 SCR for REST.MSG.Outbound Server

Item	Function	Reference	Requirement
REST-MSG-OUTB-S-001-M	Support for outbound messages	6.9	
REST-MSG-OUTB-S-002-O	Retrieve list of pending outgoing message requests - GET	6.9.3	
REST-MSG-OUTB-S-003-M	Create outgoing message request - POST (XML and JSON)	6.9.5	
REST-MSG-OUTB-S-004-O	Create outgoing message request - POST (application-x-www-form-urlencoded)	C.1	

B.1.10 SCR for REST.MSG.Outbound.MsgAndDeliveryStatus Server

Item	Function	Reference	Requirement
REST-MSG-OUTB-MSGDELSTAT-S-001-O	Support for requesting an outbound message and its delivery status	6.10	REST-MSG-OUTB-MSGDELSTAT-S-002-O
REST-MSG-OUTB-MSGDELSTAT-S-002-O	Retrieve Outgoing Message Delivery Status - GET	6.10.3	

B.1.11 SCR for REST.MSG.Outbound.DeliveryStatus Server

Item	Function	Reference	Requirement
REST-MSG-OUTB-DELSTAT-S-001-M	Support for requesting delivery status of outbound messages	6.11	
REST-MSG-OUTB-DELSTAT-S-002-M	Retrieve Outgoing Message Delivery Status - GET	6.11.3	

B.1.12 SCR for REST.MSG.Outbound.Subscriptions Server

Item	Function	Reference	Requirement
REST-MSG-OUTB-SUBSCR-S-001-M	Support for outbound subscriptions for a particular client	6.12	
REST-MSG-OUTB-SUBSCR-S-002-O	Read all outbound message delivery notification subscriptions - GET	6.12.3	
REST-MSG-OUTB-SUBSCR-S-003-M	Create new outbound message subscription – POST (XML and JSON)	6.12.5	
REST-MSG-OUTB-SUBSCR-S-003-O	Create new outbound message subscription – POST (application/x-www-form-urlencoded)	C.2	

B.1.13 SCR for REST.MSG.Individual.Outbound.Subscr Server

Item	Function	Reference	Requirement
REST-MSG-IND-OUTB-IND-SUBSCR-S-001-M	Support for outbound subscriptions for a particular client	6.13	
REST-MSG-IND-OUTB-IND-SUBSCR-S-002-O	Read individual message delivery notification subscription - GET	6.13.3	
REST-MSG-IND-OUTB-IND-SUBSCR-S-003-M	Delete subscription for the client - DELETE	6.13.6	

B.1.14 SCR for REST.MSG.Outbound.DeliveryStatus.Notifications Server

Item	Function	Reference	Requirement
REST-MSG-OUTB-DELSTAT-NOTIF-S-001-M	Support for notifying application about delivery status of outbound messages	6.14	
REST-MSG-OUTB-DELSTAT-NOTIF-S-002-M	Notify application about delivery status of outbound message - POST	6.14.5	

B.1.15 SCR for REST.MSG.Individual.Inbound.Message.Status Server

Item	Function	Reference	Requirement
REST-MSG-IND-INB-STAT-S-001-M	Support for handling of individual inbound message status	6.15	
REST-MSG-IND-INB-STAT-S-002-M	Report the status of the received inbound message - PUT	6.15.4	

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

This section defines a format for the RESTful Messaging 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 at [W3C_URLENC].

The encoding is defined below for the following Messaging REST operations which are based on POST requests:

- Sending a message to a terminal
- A mechanism to start the notification of delivery receipts
- A mechanism to start the notification of received messages

C.1 Send a message to a terminal

This operation is used to create an outgoing message request, see section 6.9.5.

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
address	xsd:anyURI [1…unbounded]	No	Destination address(es) for the message (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
senderAddress	xsd:anyURI	No	The address of the sender to whom a responding message may be sent (e.g. 'sip' URI, 'tel' URI, 'acr' URI). If senderAddress is also part of the request URL, the two MUST have the same value.
senderName	xsd:string	Yes	Name of the sender to appear on the user's terminal as the originator of the message. If this parameter is used, a set of allowed values shall be set during provisioning each sender (i.e.: for each User provisioned in the System).
chargingDescription	xsd:string [0..unbounded]	Yes	Description of charge to apply to this message. In case charging is required, this parameter MUST be present.
chargingCurrency	xsd:string	Yes	Currency of charge to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.
chargingAmount	xsd:decimal	Yes	Charging amount to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.
chargingCode	xsd:string	Yes	Charging code to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.

notifyURL	xsd:anyURI	Yes	URL to notify the application for delivery receipts. For the use of Client-side Notification URLs and Server-side Notification URLs in this parameter, see sections 6.9 and 6.9.5.
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	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
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 SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate outbound message request creation 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.
subject	xsd:string	Yes	If present, indicates the subject of the received message.
priority	MessagePriority	Yes	The priority of the message: default is Normal.

C.1.1 Example: Create outgoing message (Informative)

C.1.1.1 Request

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Date: Thu, 04 Jun 2009 02:51:59 GMT
Accept: application/xml
Host: www.example.com
Content-Type: multipart/form-data;
        boundary="==========123456==";
Content-Length: nnnn
MIME-Version: 1.0


--==========123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/x-www-form-urlencoded;
address=tel%3A%2B19585550103&
address=tel%3A%2B19585550104&
senderAddress= tel%3A%2B19585550100&
```

```
subject=hello%20from%20the%20rest%20of%20us!&
priority=High&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/77777&
callbackData=12345&
clientCorrelator=567895&
senderName=MyName
--===============123456==
Content-Disposition: form-data; name="attachments"; filename="picture.jpg"
Content-Type: image/gif

GIF89a...binary image data...
--===============123456==--
```

C.1.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550102/requests/req123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:outboundMessageRequest xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>tel:+19585550100</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</msg:outboundMessageRequest>
```

C.2 Start delivery receipt notification

This REST method is used by the application to subscribe for the delivery receipt notifications, see section 6.12.5.

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]).

Request parameters are as follows:

Name	Type/Values	Optional	Description
filterCriteria	xsd:string	No	The FilterCriteria will allow the service to filter flexibly. One example would be for the Service Provider to filter based on first 4 digits in MSISDN. This however is implementation specific and will be left to the Service Provider.
notifyURL	xsd:anyURI	No	Notification endpoint definition. For the use of Client-side Notification URLs and Server-side Notification URLs in this parameter, see sections 6.12 and 6.12.5.
callbackData	xsd:string	No	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	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
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 subscription creation 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.

C.2.1 Example: Create outbound delivery notification subscription using ‘tel’ URI (Informative)

C.2.1.1 Request

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

filterCriteria=0102&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/77777
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123
   </resourceURL>
</msg:deliveryReceiptSubscription>
```

C.2.2 Example: Create outbound delivery notification subscription using ‘acr’ URI (Informative)

C.2.2.1 Request

```
POST /exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

filterCriteria=0102&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/77777
```

C.2.2.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:deliveryReceiptSubscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123
   </resourceURL>
</msg:deliveryReceiptSubscription>
```

C.3 Start message notification

This REST method is used by the application to subscribe for the notifications of received messages, see section 6.6.5.

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]).

Request parameters are as follows:

Name	Type/Values	Optional	Description
destinationAddress	xsd:anyURI[1..unbounded]	No	Destination address of the message (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
criteria	xsd:string	Yes	The text to match against to determine the application to receive the notification
notifyURL	xsd:anyURI	No	Notification endpoint definition. For the use of Client-side Notification URLs and Server-side Notification URLs in this parameter, see sections 6.6 and 6.6.5.
callbackData	xsd:string	No	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	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
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 subscription creation 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.
useAttachmentURLs	xsd:boolean	Yes	Default: false If set to 'true', inbound message would have links to attachments together with the indication of the content type and optionally the size of each attachment. Otherwise, only message identifier will be returned, so that individual message retrieval can be done

This operation would return a result indicating whether the operation has been successful.

C.3.1 Example: Create inbound subscription (Informative)

C.3.1.1 Request

```
POST /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

destinationAddress=tel:+19585550100&
criteria=Urgent*&
clientCorrelator=567893&
useAttachmentURLs=false&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/88888&
callbackData=12345&
notificationFormat=XML
```

C.3.1.2 Response

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<msg:subscription xmlns:msg="urn:oma:xml:rest:netapi:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification//88888</notifyURL>
    <callbackData>12345</callbackData>
    <notificationFormat>XML</notificationFormat>  </callbackReference>
  <destinationAddress>tel:+19585550100</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567893</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123</resourceURL>
  <useAttachmentURLs>false</useAttachmentURLs>
</msg:subscription>
```


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 invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC7159].

The following examples show the request and response for various operations using a JSON binding. 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 Retrieve messages for a registration (section 6.1.3.1)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"inboundMessageList": {
   "inboundMessage": [
      {
         "destinationAddress": "tel:+19585550100",
         "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
         "messageId": "msg123",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
         "senderAddress": "tel:+19585550101"
      },
      {
         "destinationAddress": "tel:+19585550102",
         "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
         "messageId": "msg124",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg124",
         "senderAddress": "tel:+19585550103"
      }
   ],
   "numberOfMessagesInThisBatch": "2",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2",
   "totalNumberOfPendingMessages": "20"
}}
```

D.2 Request with invalid (non-existing) id (section 6.1.3.2)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 404 Not Found
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"requestError": {
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2",
      "rel": "self"
   },
   "serviceException": {
      "messageId": "SVC0004",
      "text": "No valid addresses provided in message part %1",
      "variables": "reg123"
   }
}}
```

D.3 Retrieve messages with attachment URLs (section 6.1.3.3)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=2&useAttachmentURLs=true HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"inboundMessageList": {
   "inboundMessage": {
      "destinationAddress": "tel:+19585550100",
      "inboundMMSMessage": {
         "attachment": [
            {
               "contentType": "image/gif",
               "link": {
                  "href":
"http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach123",
                  "rel": "attachment"
               },
```

```
                "size": "27000"
            },
            {
                "contentType": "video/3gpp",
                "link": {
                    "href":
"http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach124",
                    "rel": "attachment"
                },
                "size": "125000"
            }
        ],
        "bodyText": "See attached picture",
        "subject": "Who is RESTing on the beach?"
    },
    "messageId": "msg123",
    "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
    "senderAddress": "tel:+19585550101"
  },
  "numberOfMessagesInThisBatch": "2",
  "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/msg123/attachments",
  "totalNumberOfPendingMessages": "20"
}}
```


D.4 MaxBatchSize exceeding the allowed size (section 6.1.3.4)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=5000 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 403 Forbidden
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"requestError": {
  "link": {
    "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages?maxBatchSize=5000",
    "rel": "InboundMessageList"
  },
  "policyException": {
    "messageId": "POL1020",
    "text": "MaxBatchSize exceeded. The maximum allowed maxBatchSize is %1.",
    "variables": "20"
  }
}}
```


D.5 Retrieve and delete inbound messages (section 6.2.5.1)

Request:

```
POST /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/retrieveAndDeleteMessages  HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: application/json
Content-Length: nnnn

{"inboundMessageRetrieveAndDeleteRequest": {
   "retrievalOrder": "OldestFirst",
   "useAttachmentURLs": "false"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"inboundMessageList": {
   "inboundMessage": [
      {
         "destinationAddress": "tel:+19585550100",
         "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
         "messageId": "msg123",
         "senderAddress": "tel:+19585550101"
      },
      {
         "destinationAddress": "tel:+19585550102",
         "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
         "messageId": "msg124",
         "senderAddress": "tel:+19585550103"
      }
   ],
   "numberOfMessagesInThisBatch": "2",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/retrieveAndDeleteMessages",
   "totalNumberOfPendingMessages": "20"
}}
```

D.6 Read and delete one message (section 6.3.5.1)

Request:

```
POST /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/retrieveAndDelete HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"inboundMessageRetrieveAndDeleteRequest": {"useAttachmentURLs": "false"}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Length: nnnn
Content-Type: multipart/form-data;,                boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name="root-fields"
Content-Type: application/json

{"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
   "messageId": "msg123",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}

--===============123456==
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment;filename="textBody.txt";
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit

Look at the attached picture

--====aaabbb
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>

GIF89a...binary image data...

--====aaabbb--
```

```
--===============123456==--
```

D.7 Read message from gateway storage (section 6.4.3.1)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123?resFormat=JSON HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: multipart/form-data; boundary="=====12345===="
Content-Length: nnnn

====12345====
Content-Disposition=multipart/form-data; name="root-fields"
Content-Type=application/json
Content-Length: nnnn

{"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
   "messageId": "msg123",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}

--====12345====
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment;filename="textBody.txt";
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit


Look at the attached picture


--====aaabbb
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>

GIF89a...binary image data...
--====aaabbb--
--===============123456==--
```

D.8 Read message from gateway storage, Displayed status report requested (section 6.4.3.2)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: multipart/form-data; boundary="=====12345===="
Content-Length: nnnn

====12345====
Content-Disposition=multipart/form-data; name="root-fields"
Content-Type=application/json
Content-Length: nnnn

{"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status",
      "rel": "MessageStatusReport"
   },
   "messageId": "msg123",
   "reportRequest": "Displayed",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}

--====12345====
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment;filename="textBody.txt";
Content-Type: text/plain
Content-Transfer-Encoding: 8 bit

Look at the attached picture

--====aaabbb
Content-Disposition:attachment;filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
Content-ID: <99334422@example.com>

GIF89a...binary image data...
--====aaabbb--
```

```
--===============123456==--
```

D.9 Remove message from gateway storage (section 6.4.6.1)

Request:

```
DELETE /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.10 Read an MMS attachment (section 6.5.3.1)

Request:

```
GET /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: image/gif, image/png, image/jpeg, text/html, application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/gif
Content-Length: nnnn

...GIF89a...binary image data
```

D.11 Delete an MMS attachment from gateway storage (section 6.5.6.1)

Request:

```
DELETE /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.12 Read active subscriptions (section 6.6.3.1)

Request:

```
GET /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"subscriptionList": {
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions",
   "subscription": [
      {
         "callbackReference": {
            "callbackData": "12345",
            "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/12345"
         },
         "clientCorrelator": "567891",
         "criteria": "Urgent*",
         "destinationAddress": "tel:+19585550101",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/0000001",
         "useAttachmentURLs": "false"
      },
      {
         "callbackReference": {
            "callbackData": "54321",
            "notificationFormat": "XML",
            "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/54321"
         },
         "clientCorrelator": "567892",
         "criteria": "Urgent*",
         "destinationAddress": "tel:+19585550102",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/0000002",
         "useAttachmentURLs": "false"
      }
   ]
}}
```

D.13 Create inbound subscription (returning a representation of created resource) (section 6.6.5.1)

Request:

```
POST /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"subscription": {
   "callbackReference": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
   },
   "clientCorrelator": "567893",
   "criteria": "Urgent*",
   "destinationAddress": "tel:+19585550100",
   "useAttachmentURLs": "false"
}}
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123
Content-Type: application/json
Content-Length: nnnn

{"subscription": {
   "callbackReference": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
   },
   "clientCorrelator": "567893",
   "criteria": "Urgent*",
   "destinationAddress": "tel:+19585550100",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123",
   "useAttachmentURLs": "false"
}}
```

D.14 Create inbound subscription (returning location of created resource) (section 6.6.5.2)

Request:

```
POST /exampleAPI/messaging/v1/inbound/subscriptions HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"subscription": {
   "callbackReference": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
   },
   "criteria": "Urgent*",
   "destinationAddress": "tel:+19585550100",
   "useAttachmentURLs": "false"
}}
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123
Content-Type: application/json
Content-Length: nnnn

{"resourceReference": {"resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123"}}
```

D.15 Read individual subscription (section 6.7.3.1)

Request:

```
GET /exampleAPI/messaging/v1/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"subscription": {
   "callbackReference": {
      "notificationFormat": "XML",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
   },
   "clientCorrelator": "567893",
```

```
   "criteria": "Urgent*",
   "destinationAddress": "tel:+19585550100",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123",
   "useAttachmentURLs": "false"
}}
```


D.16 Delete a subscription (section 6.7.6.1)

Request:

```
DELETE /exampleAPI/messaging/v1/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.17 Message arrival notification (section 6.8.5.1)

Request:

```
POST /notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"inboundMessageNotification": {"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123",
      "rel": "Subscription"
   },
   "messageId": "msg123",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```


D.18 Message arrival notification with attachment URLs (section 6.8.5.2)

Request:

```
POST {HYPERLINK "../../application.example.com/notifications/DeliveryInfoNotification"}/88888 HTTP/1.1
Accept: application/json
Host: application.example.com
Content-Type: application/json
Content-Length: nnnn

{"inboundMessageNotification": {"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {
      "attachment": [
         {
            "contentType": "image/gif",
            "link": {
               "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123
/attachments/attach123",
               "rel": "attachment"
            },
            "size": "27000"
         },
         {
            "contentType": "video/3gpp",
            "link": {
               "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/
attachments/attach124",
               "rel": "attachment"
            },
            "size": "125000"
         }
      ],
      "bodyText": "Look at the attached picture",
      "subject": "Who is RESTing on the beach?"
   },
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123",
      "rel": "Subscription"
   },
   "messageId": "msg123",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.19 Message (with Displayed status report requested) arrival notification (section 6.8.5.3)

Request:

```
POST /notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: example.com
Content-Length: nnnn

{"inboundMessageNotification": {"inboundMessage": {
   "destinationAddress": "tel:+19585550100",
   "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
   "link": [
      {
         "href": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status",
         "rel": "MessageStatusReport"
      },
      {
         "href": "http://example.com/exampleAPI/messaging/v1/inbound/subscriptions/sub123",
         "rel": "Subscription"
      }
   ],
   "messageId": "msg123",
   "reportRequest": "Displayed",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123",
   "senderAddress": "tel:+19585550101"
}}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.20 Retrieve list of outgoing requests (section 6.9.3.1)

Request:

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

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequestList": {
   "outboundMessageRequest": {
      "address": "tel:+19585550103",
      "clientCorrelator": "567894",
```

```
      "deliveryInfoList": {
         "deliveryInfo": {
            "address": "tel:+19585550103",
            "deliveryStatus": "DeliveredToTerminal"
         },
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos"
      },
      "outboundMMSMessage": {"subject": "Holiday greetings"},
      "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
      "senderAddress": "tel:+19585550100"
   },
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests"
}}
```

D.21 Create outgoing message, returning the representation of created resource (section 6.9.5.1)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
```

```
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
"address": [
"tel:+19585550103",
"tel:+19585550104"
],
"clientCorrelator": "567895",
"outboundMMSMessage": {
"priority": "High",
"subject": "hello from the rest of us!"
},
"receiptRequest": {
"callbackData": "12345",
"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
},
"resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
"senderAddress": "tel:+19585550100",
"senderName": "MyName"
}}
```

D.22 Create outgoing message, returning the location of created resource (section 6.9.5.2)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/json
Content-Length: nnnn

{"resourceReference": {"resourceURL": "
http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123"}}
```

D.23 Create outgoing message with charging (section 6.9.5.3)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: form-data; name="root-fields"
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "charging": {"description": "Sample text for the charging information"},
   "clientCorrelator": "567896",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==
Content-Disposition: multipart/form-data; name="attachments"
Content-Type: multipart/mixed; boundary="===12345==="

--===12345===
Content-Disposition: attachment; filename="text.txt"
Content-Type: text/plain
Content-Length: nnnn

See attached photo
```

```
--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif
Content-Length: nnnn

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response for charging not supported:

```
HTTP/1.1 400 Bad request
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"requestError": {"policyException": {
   "messageId": "POL0008",
   "text": "Charging is not supported"
}}}
```

D.24 Create outgoing message, serviceException in case of address(es) failure (section 6.9.5.4)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
```

```
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"requestError": {"serviceException": {
   "messageId": "SVC0004",
   "text": "No valid addresses provided in message part %1",
   "variables": "address"
}}}
```

D.25 Create outgoing message, multiple addresses partial success, with deliveryInfoList in response (section 6.9.5.5)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==
Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "deliveryInfoList": {
      "deliveryInfo": [
         {
            "address": "tel:+19585550103",
            "deliveryStatus": "MessageWaiting"
         },
         {
            "address": "tel:+19585550104",
            "deliveryStatus": "DeliveryImpossible"
         }
      ],
      "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos"
},
   "outboundMMSMessage": {
      "priority": "High",
      "subject": " hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}
```

D.26 Create outgoing message, multiple addresses partial success, without deliveryInfoList in response (section 6.9.5.6)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": "hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
      "priority": "High",
      "subject": " hello from the rest of us!"
   },
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
   "senderAddress": " tel:+19585550100",
   "senderName": "MyName"
}}
```

D.27 Create outgoing message using SHORT CODE as senderAddress, returning the representation of created resource (section 6.9.5.7)

Request:

```
POST /exampleAPI/messaging/v1/outbound/72654/requests HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: multipart/form-data; boundary="===============123456==";
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "outboundMMSMessage": {
                                                      "priority": "High",
                                                "subject": "hello from the rest of us!"
```

```
},
"receiptRequest": {
"callbackData": "12345",
"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
},
"senderAddress": "72654",
"senderName": "MyName"
}}

--===============123456==

Content-Disposition: multipart/form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”===12345===”

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: text/plain;

See attached photo

--===12345===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...

--===12345===--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/72654/requests/req123
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
"address": [
"tel:+19585550103",
"tel:+19585550104"
],
"clientCorrelator": "567895",
"outboundMMSMessage": {
"priority": "High",
"subject": "hello from the rest of us!"
},
"receiptRequest": {
"callbackData": "12345",
"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
},
"resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/72654/requests/req123",
"senderAddress": "72654",
"senderName": "MyName"
}}
```

D.28 Read message request and delivery status (section 6.10.3.1)

Request:

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"outboundMessageRequest": {
   "address": [
      "tel:+19585550103",
      "tel:+19585550104"
   ],
   "clientCorrelator": "567895",
   "deliveryInfoList": {
      "deliveryInfo": [
         {
            "address": "tel:+19585550103",
            "deliveryStatus": "MessageWaiting"
         },
         {
            "address": "tel:+19585550104",
            "deliveryStatus": "MessageWaiting"
         }
      ],
      "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfos"
   },
   "outboundMMSMessage": {"subject": "Holiday greetings"},
   "receiptRequest": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
   "senderAddress": "tel:+19585550100",
   "senderName": "MyName"
}}
```

D.29 Read message delivery status (section 6.11.3.1)

Request:

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"deliveryInfoList": {
   "deliveryInfo": [
      {
         "address": "tel:+19585550103",
         "deliveryStatus": "MessageWaiting"
      },
      {
         "address": "tel:+19585550104",
         "deliveryStatus": "MessageWaiting"
      }
   ],
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos"
}}
```

D.30 Read delivery notification subscriptions (section 6.12.3.1)

Request:

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

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"deliveryReceiptSubscriptionList": {
   "deliveryReceiptSubscription": [
      {
         "callbackReference": {
            "callbackData": "12345",
            "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
         },
         "filterCriteria": "0102",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123"
      },
      {
         "callbackReference": {
            "callbackData": "54321",
            "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77778"
         },
         "filterCriteria": "0103",
         "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123"
      }
   ],
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions"
}}
```

D.31 Create outbound delivery notification subscription using ‘tel’ URI (section 6.12.5.1)

Request:

```
POST /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: application/json
Content-Length: nnnn{"deliveryReceiptSubscription": {
   "callbackReference": {"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"},
   "filterCriteria": "0102"
}}
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123
Content-Type: application/json
Content-Length: nnnn


{"deliveryReceiptSubscription": {
   "callbackReference": {"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"},
   "filterCriteria": "0102",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123"
}}
```

D.32 Create outbound delivery notification subscription using ‘acr’ URI (section 6.12.5.2)

Request:

```
POST /exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
Content-Type: application/json
Content-Length: nnnn

{"deliveryReceiptSubscription": {
   "callbackReference": {"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"},
   "filterCriteria": "0102"
}}
```

Response:

```
HTTP/1.1 201 Created
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123
Content-Type: application/json
Content-Length: nnnn

{"deliveryReceiptSubscription": {
   "callbackReference": {"notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"},
   "filterCriteria": "0102",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/acr%3Apseudonym123/subscriptions/sub123"
}}
```

D.33 Read individual message delivery notification subscription (section 6.13.3.1)

Request:

```
GET /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"deliveryReceiptSubscription": {
   "callbackReference": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
   },
   "filterCriteria": "0102",
   "resourceURL": "http://example.com/exampleAPI/messaging/v1/outbound/ tel%3A%2B19585550100/subscriptions/sub123"
}}
```

D.34 Delete message delivery notification subscription (section 6.13.6.1)

Request:

```
DELETE /exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.35 Notify client about outbound message delivery status, multiple delivery status per notification (section 6.14.5.1)

Request:

```
POST /notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/json
Host: application.example.com
Content-Type: application/json
Content-Length: nnnn

{"deliveryInfoNotification": {
   "deliveryInfo": [
      {
         "address": "tel:+19585550103",
         "deliveryStatus": "DeliveredToTerminal"
      },
      {
         "address": "tel:+19585550104",
         "deliveryStatus": "DeliveredToTerminal"
      }
   ],
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
      "rel": "OutboundMessageRequest"
   }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.36 Notify client about outbound message delivery status, single delivery status per notification (section 6.14.5.2)

Request:

```
POST /notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/json
Host: application.example.com
Content-Type: application/json
Content-Length: nnnn

{"deliveryInfoNotification": {
   "deliveryInfo": {
      "address": "tel:+19585550104",
      "deliveryStatus": "DeliveredToTerminal"
   },
   "link": {
      "href": "http://example.com/exampleAPI/messaging/v1/outbound/tel%3A%2B19585550100/requests/req123",
      "rel": "OutboundMessageRequest"
   }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.37 Reporting the status of an inbound message (section 6.15.4.1)

Request:

```
PUT /exampleAPI/messaging/v1/inbound/registrations/reg123/messages/msg123/status HTTP/1.1
Accept: application/json
Host: application.example.com
Content-Type: application/json
Content-Length: nnnn

{"messageStatusReport": {"status": "Displayed"}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

Appendix E. Parlay X operations mapping (Informative)

The table below illustrates the mapping between REST resources/methods and Parlay X [3GPP 29.199-5] equivalent operations.

REST Resource	REST Method	REST Section reference	Parlay X equivalent operation
Inbound messages for a given registration	GET	6.1.3	getReceivedMessages 1)
Inbound messages retrieve and delete using registration	POST	6.2.5	getReceivedMessages
Retrieval and deletion of individual inbound message using registration	POST	6.3.5	getMessage
Inbound message for a given registration	GET	6.4.3	getMessage, getMessageURIs 2)
	DELETE	6.4.6	getReceivedMessages
Inbound message subscriptions	POST	6.6.5	startMessageNotification
Individual inbound message subscription	DELETE	6.7.6	stopMessageNotification
Client notification about inbound message	POST	6.8.5	notifyMessageReception
Outbound message requests	POST	6.9.5	sendMessage
Outbound message delivery status	GET	6.11.3	getMessageDeliveryStatus
Outbound message delivery notification subscriptions	POST	6.12.5	startDeliveryReceiptNotification
Individual outbound message delivery notification subscription	DELETE	6.13.6	stopDeliveryReceiptNotification
Client notification about outbound message delivery status	POST	6.14.5	notifyMessageDeliveryReceipt

Table 1 Parlay X operations mapping

1) Note: The ParlayX SOAP operation getReceivedMessages is similar to but not quite the same as this REST method because DELETE of individual message is required for confirmation of successful retrieval (see DELETE on Inbound message).

2) In case that parameter “useAttachmentURLs” is set to “true”, equivalent ParlayX SOAP operation is getMessageURIs.

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 Messaging API in combination with some authorization frameworks.

G.1 Use with OMA Authorization Framework for Network APIs

The RESTful Messaging API MAY support the authorization framework defined in [Autho4API_10].

A RESTful Messaging 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 Messaging 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_messaging.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 next rows of this table.	No
oma_rest_messaging.in_regist	Provide access to all defined operations on inbound messages using registration	No
oma_rest_messaging.in_subscr	Provide access to all defined operations on inbound messages using subscription	No
oma_rest_messaging.out	Provide access to all defined operations on outbound messages	No

Table 2 Scope values for RESTful Messaging API

G.1.1.2 Downscoping

In the case where the client requests authorization for “oma_rest_messaging.all_{apiVersion}” scope, the authorization server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- “oma_rest_messaging.in_regist”
- “oma_rest_messaging.in_subscr”
- “oma_rest_messaging.out”

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 Messaging API map to the REST resources and methods of this API. In these tables, the root “oma_rest_messaging.” of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Inbound messages for a given registration	/inbound/registrations/{registrationId}/messages	6.1	**all_{apiVersion}** or **in_regist**	n/a	n/a	n/a
Inbound messages retrieve and delete using registration	/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages	6.2	n/a	n/a	**all_{apiVersion}** or **in_regist**	n/a
Retrieval and deletion of individual inbound message using registration	/inbound/registrations/{registrationId}/messages/{messageId}/retrieveAndDelete	6.3	n/a	n/a	**all_{apiVersion}** or **in_regist**	n/a
Inbound message for a given registration	/inbound/registrations/{registrationId}/messages/{messageId}	6.4	**all_{apiVersion}** or **in_regist**	n/a	n/a	**all_{apiVersion}** or **in_regist**
Inbound message attachment	/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId}	6.5	**all_{apiVersion}** or **in_regist**	n/a	n/a	**all_{apiVersion}** or **in_regist**

Table 3 Required scope values for: inbound messages for periodic polling

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Inbound message subscriptions	/inbound/subscriptions	6.6	**all_{apiVersion}** or **in_subscr**	n/a	**all_{apiVersion}** or **in_subscr**	n/a
Individual inbound message subscription	/inbound/subscriptions/{subscriptionId}	6.7	**all_{apiVersion}** or **in_subscr**	n/a	n/a	**all_{apiVersion}** or **in_subscr**

Table 4 Required scope values for: subscription management for inbound messages

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Outbound message requests	/outbound/{senderAddress}/requests	6.9	**all_{apiVersion}** or **out**	n/a	**all_{apiVersion}** or **out**	n/a
Outbound message request and delivery status	/outbound/{senderAddress}/requests/{requestId}	6.10	**all_{apiVersion}** or **out**	n/a	n/a	n/a
Outbound message delivery status	/outbound/{senderAddress}/requests/{requestId}/deliveryInfos	6.11	**all_{apiVersion}** or **out**	n/a	n/a	n/a

Table 5 Required scope values for: sending message and obtaining the delivery status

Resource	URL Base URL: http://{serverRoot}/messaging/{apiVersion}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Outbound message delivery notification subscriptions	/outbound/{senderAddress}/subscriptions	6.12	**all_{apiVersion}** or **out**	n/a	**all_{apiVersion}** or **out**	n/a
Individual outbound message delivery notification subscription	/outbound/{senderAddress}/subscriptions/{subscriptionId}	6.13	**all_{apiVersion}** or **out**	n/a	n/a	**all_{apiVersion}** or **out**

Table 6 Required scope values for: subscription management for outbound message delivery status

Resource	URL <specified by the server>	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Individual inbound message status	Specified by the server when the inbound message is delivered to the client	6.15	n/a	**all_{apiVersion}** or **in_regist** or **in_subscr**	n/a	n/a

Table 7 Required scope values for: notifying the server about inbound message status

G.1.2 Use of ‘acr:auth’

This section specifies the use of ‘acr:auth’ in place of an end user identifier in a resource URL path.

An ‘acr’ URI of the form ‘acr:auth’, where ‘auth’ 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:auth’ in a resource URL in place of the{senderAddress} resource URL variable in the resource URL path, when the RESTful Messaging API is used in combination with [Autho4API_10].

In the case the RESTful Messaging API supports [Autho4API_10], the server:

– SHALL accept ‘acr:auth’ as a valid value for the resource URL variable {senderAddress}.

– SHALL conform to [REST_NetAPI_Common] section 5.8.1.1 regarding the processing of ‘acr:auth’.