



RESTful bindings for Parlay X Web Services – Multi-media Messaging

Approved Version 1.1 – 24 Jul 2012

Open Mobile Alliance
OMA-TS-ParlayREST_MultiMediaMessaging-V1_1-20120724-A

Use of this document is subject to all of the terms and conditions of the Use Agreement located at <http://www.openmobilealliance.org/UseAgreement.html>.

Unless this document is clearly designated as an approved specification, this document is a work in process, is not an approved Open Mobile Alliance™ specification, and is subject to revision or removal without notice.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. Information contained in this document may be used, at your sole risk, for any purposes. You may not use this document in any other manner without the prior written permission of the Open Mobile Alliance. The Open Mobile Alliance authorizes you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services. The Open Mobile Alliance assumes no responsibility for errors or omissions in this document.

Each Open Mobile Alliance member has agreed to use reasonable endeavors to inform the Open Mobile Alliance in a timely manner of Essential IPR as it becomes aware that the Essential IPR is related to the prepared or published specification. However, the members do not have an obligation to conduct IPR searches. The declared Essential IPR is publicly available to members and non-members of the Open Mobile Alliance and may be found on the “OMA IPR Declarations” list at <http://www.openmobilealliance.org/ipr.html>. The Open Mobile Alliance has not conducted an independent IPR review of this document and the information contained herein, and makes no representations or warranties regarding third party IPR, including without limitation patents, copyrights or trade secret rights. This document may contain inventions for which you must obtain licenses from third parties before making, using or selling the inventions. Defined terms above are set forth in the schedule to the Open Mobile Alliance Application Form.

NO REPRESENTATIONS OR WARRANTIES (WHETHER EXPRESS OR IMPLIED) ARE MADE BY THE OPEN MOBILE ALLIANCE OR ANY OPEN MOBILE ALLIANCE MEMBER OR ITS AFFILIATES REGARDING ANY OF THE IPR'S REPRESENTED ON THE “OMA IPR DECLARATIONS” LIST, INCLUDING, BUT NOT LIMITED TO THE ACCURACY, COMPLETENESS, VALIDITY OR RELEVANCE OF THE INFORMATION OR WHETHER OR NOT SUCH RIGHTS ARE ESSENTIAL OR NON-ESSENTIAL.

THE OPEN MOBILE ALLIANCE IS NOT LIABLE FOR AND HEREBY DISCLAIMS ANY DIRECT, INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR EXEMPLARY DAMAGES ARISING OUT OF OR IN CONNECTION WITH THE USE OF DOCUMENTS AND THE INFORMATION CONTAINED IN THE DOCUMENTS.

© 2012 Open Mobile Alliance Ltd. All Rights Reserved.

Used with the permission of the Open Mobile Alliance Ltd. under the terms set forth above.

Contents

1.	SCOPE	8
2.	REFERENCES	9
2.1	NORMATIVE REFERENCES	9
2.2	INFORMATIVE REFERENCES	9
3.	TERMINOLOGY AND CONVENTIONS	10
3.1	CONVENTIONS	10
3.2	DEFINITIONS	10
3.3	ABBREVIATIONS	10
4.	INTRODUCTION	11
4.1	VERSION 1.0	11
4.2	VERSION 1.1	11
5.	MESSAGING API DEFINITION	12
5.1	RESOURCES SUMMARY	13
5.2	MESSAGING (MMS, SMS, WAP, IM) PARLAYREST API DATA STRUCTURES	18
5.2.1	Type: InboundMessageList.....	18
5.2.2	Type: InboundMessage	18
5.2.3	Type: InboundMessageNotification.....	19
5.2.4	Type: InboundSMSTextMessage.....	20
5.2.5	Type: InboundMMSMessage.....	20
5.2.6	Type: InboundIMMessage	20
5.2.7	Type: SubscriptionList.....	21
5.2.8	Type: Subscription	21
5.2.9	Type: InboundMessageRetrieveAndDeleteRequest.....	22
5.2.10	Type: OutboundMessageRequestList	23
5.2.11	Type: OutboundMessageRequest	23
5.2.12	Type: OutboundMMSMessage	25
5.2.13	Type: OutboundWAPMessage	25
5.2.14	Type: OutboundSMSTextMessage	26
5.2.15	Type: OutboundSMSBinaryMessage.....	27
5.2.16	Type: OutboundSMSLogoMessage	27
5.2.17	Type: OutboundSMSRingToneMessage	27
5.2.18	Type: OutboundIMMessage	27
5.2.19	Type: DeliveryInfoList	28
5.2.20	Type: DeliveryInfoNotification	28
5.2.21	Type: DeliveryInfo.....	28
5.2.22	Type: DeliveryReceiptSubscriptionList.....	29
5.2.23	Type: DeliveryReceiptSubscription	30
5.2.24	Enumeration: DeliveryStatus	30
5.2.25	Enumeration: IMFormat	31
5.2.26	Enumeration: MessagePriority.....	31
5.2.27	Enumeration: RetrievalOrder.....	31
5.2.28	Enumeration: ServiceIndicationAction	31
5.2.29	Enumeration: ServiceLoadingAction.....	32
5.2.30	Enumeration: SmsFormat	32
5.2.31	Enumeration: WAPContent	32
5.2.32	Values of the Link “rel” attribute.....	33
5.2.33	MIME multipart representation	33
5.3	SEQUENCE DIAGRAMS	33
5.3.1	Send message and check the delivery status	33
5.3.2	Inbound message delivery (push mode).....	35
5.3.3	Inbound message delivery (polling mode).....	36
5.4	RESOURCE: INBOUND MESSAGES FOR A GIVEN REGISTRATION	37

5.4.1	Request URI variables	38
5.4.2	Response Codes	38
5.4.3	GET.....	38
5.4.4	PUT.....	41
5.4.5	POST.....	41
5.4.6	DELETE	41
5.5	RESOURCE: INBOUND MESSAGES RETRIEVE AND DELETE USING REGISTRATION	41
5.5.1	Request URI variables	41
5.5.2	Response Codes	41
5.5.3	GET.....	42
5.5.4	PUT.....	42
5.5.5	POST.....	42
5.5.6	DELETE	43
5.6	RESOURCE: RETRIEVAL AND DELETION OF INDIVIDUAL INBOUND MESSAGE USING REGISTRATION	43
5.6.1	Request URI variables	43
5.6.2	Response Codes	44
5.6.3	GET.....	44
5.6.4	PUT.....	44
5.6.5	POST.....	44
5.6.6	DELETE	45
5.7	RESOURCE: INBOUND MESSAGE FOR A GIVEN REGISTRATION	45
5.7.1	Request URI variables	46
5.7.2	Response Codes	46
5.7.3	GET.....	46
5.7.4	PUT.....	47
5.7.5	POST.....	47
5.7.6	DELETE	48
5.8	RESOURCE: INBOUND MESSAGE ATTACHMENT	48
5.8.1	Request URI variables	48
5.8.2	Response Codes	49
5.8.3	GET.....	49
5.8.4	PUT.....	49
5.8.5	POST.....	49
5.8.6	DELETE	49
5.9	RESOURCE: INBOUND MESSAGE SUBSCRIPTIONS.....	50
5.9.1	Request URI variables	50
5.9.2	Response Codes	50
5.9.3	GET.....	50
5.9.4	PUT.....	51
5.9.5	POST.....	51
5.9.6	DELETE	53
5.10	RESOURCE: INDIVIDUAL INBOUND MESSAGE SUBSCRIPTION	53
5.10.1	Request URI variables	53
5.10.2	Response Codes	53
5.10.3	GET.....	53
5.10.4	PUT.....	54
5.10.5	POST.....	54
5.10.6	DELETE	54
5.11	RESOURCE: CLIENT NOTIFICATION ABOUT INBOUND MESSAGE	55
5.11.1	Request URI variables	55
5.11.2	Response Codes	55
5.11.3	GET.....	55
5.11.4	PUT.....	55
5.11.5	POST.....	55
5.11.6	DELETE	57
5.12	RESOURCE: OUTBOUND MESSAGE REQUESTS.....	57
5.12.1	Request URI variables	57

5.12.2	Response Codes	57
5.12.3	GET.....	57
5.12.4	PUT.....	58
5.12.5	POST.....	58
5.12.6	DELETE	68
5.13	RESOURCE: OUTBOUND MESSAGE REQUEST AND DELIVERY STATUS	68
5.13.1	Request URI variables	69
5.13.2	Response Codes	69
5.13.3	GET.....	69
5.13.4	PUT.....	70
5.13.5	POST.....	70
5.13.6	DELETE	70
5.14	RESOURCE: OUTBOUND MESSAGE DELIVERY STATUS.....	70
5.14.1	Request URI variables	70
5.14.2	Response Codes	71
5.14.3	GET.....	71
5.14.4	PUT.....	72
5.14.5	POST.....	72
5.14.6	DELETE	72
5.15	RESOURCE: OUTBOUND MESSAGE DELIVERY NOTIFICATION SUBSCRIPTIONS	72
5.15.1	Request URI variables	72
5.15.2	Response Codes	72
5.15.3	GET.....	72
5.15.4	PUT.....	73
5.15.5	POST.....	73
5.15.6	DELETE	74
5.16	RESOURCE: INDIVIDUAL OUTBOUND MESSAGE DELIVERY NOTIFICATION SUBSCRIPTION.....	74
5.16.1	Request URI variables	75
5.16.2	Response Codes	75
5.16.3	GET.....	75
5.16.4	PUT.....	76
5.16.5	POST.....	76
5.16.6	DELETE	76
5.17	RESOURCE: CLIENT NOTIFICATION ABOUT OUTBOUND MESSAGE DELIVERY STATUS	76
5.17.1	Request URI variables	76
5.17.2	Response Codes	76
5.17.3	GET.....	77
5.17.4	PUT.....	77
5.17.5	POST.....	77
5.17.6	DELETE	78
APPENDIX A.	CHANGE HISTORY (INFORMATIVE).....	79
A.1	APPROVED VERSION HISTORY	79
APPENDIX B.	STATIC CONFORMANCE REQUIREMENTS (NORMATIVE).....	80
B.1	SCR FOR PARLAYREST.MMS SERVER	80
B.1.1	SCR for ParlayREST.MMS.Inbound.Registration Server	80
B.1.2	SCR for ParlayREST.MMS.Inbound.Registration.RetrieveDelete Server	80
B.1.3	SCR for ParlayREST.MMS.Individual.Inbound.Registration.RetrieveDelete Server	80
B.1.4	SCR for ParlayREST.MMS.Individual.Inbound Server	81
B.1.5	SCR for ParlayREST.MMS.Attach.Individual.Inbound Server.....	81
B.1.6	SCR for ParlayREST.MMS.Inbound.Subscr Server.....	81
B.1.7	SCR for ParlayREST.MMS.Inbound.Individual.Subscr Server	81
B.1.8	SCR for ParlayREST.MMS.Inbound.Notifications Server.....	82
B.1.9	SCR for ParlayREST.MMS.Outbound Server.....	82
B.1.10	SCR for ParlayREST.MMS.Outbound.MsgAndDeliveryStatus Server	82
B.1.11	SCR for ParlayREST.MMS.Outbound.DeliveryStatus Server	83
B.1.12	SCR for ParlayREST.MMS.Outbound.Subscriptions Server	83

B.1.13 SCR for ParlayREST.MMS.Individual.Outbound.Subscr Server..... 83

B.1.14 SCR for ParlayREST.MMS.Outbound.DeliveryStatus.Notifications Server..... 83

APPENDIX C. APPLICATION/X-WWW-FORM-URLENCODED REQUEST FORMAT FOR SELECTED REST OPERATIONS (NORMATIVE) 84

C.1 SEND A MESSAGE TO A TERMINAL 84

C.1.1 Example: Create outgoing message (Informative)..... 85

C.2 START DELIVERY RECEIPT NOTIFICATION..... 86

C.2.1 Example: Create outbound delivery notification subscription (Informative)..... 87

C.3 START MESSAGE NOTIFICATION 88

C.3.1 Example: Create inbound subscription (Informative)..... 89

APPENDIX D. JSON EXAMPLES (INFORMATIVE) 90

D.1 RETRIEVE MESSAGES FOR A REGISTRATION (SECTION 5.4.3.1) 90

D.2 REQUEST WITH INVALID (NON-EXISTING) ID (SECTION 5.4.3.2) 91

D.3 RETRIEVE MESSAGES WITH ATTACHMENT URLS (SECTION 5.4.3.3)..... 91

D.4 RETRIEVE AND DELETE INBOUND MESSAGES (SECTION 5.5.5.1) 92

D.5 READ AND DELETE ONE MESSAGE (SECTION 5.6.5.1)..... 93

D.6 READ MESSAGE FROM GATEWAY STORAGE (SECTION 5.7.3.1)..... 94

D.7 REMOVE MESSAGE FROM GATEWAY STORAGE (SECTION 5.7.6.1) 95

D.8 READ AN MMS ATTACHMENT (SECTION 5.8.3.1) 95

D.9 DELETE AN MMS ATTACHMENT FROM GATEWAY STORAGE (SECTION 5.8.6.1) 95

D.10 READ ACTIVE SUBSCRIPTIONS (SECTION 5.9.3.1)..... 96

D.11 CREATE INBOUND SUBSCRIPTION (RETURNING A REPRESENTATION OF CREATED RESOURCE) (SECTION 5.9.5.1) 96

D.12 CREATE INBOUND SUBSCRIPTION (RETURNING LOCATION OF CREATED RESOURCE) (SECTION 5.9.5.2)..... 97

D.13 READ INDIVIDUAL SUBSCRIPTION (SECTION 5.10.3.1) 98

D.14 DELETE A SUBSCRIPTION (SECTION 5.10.6.1)..... 98

D.15 MESSAGE ARRIVAL NOTIFICATION (SECTION 5.11.5.1) 99

D.16 MESSAGE ARRIVAL NOTIFICATION WITH ATTACHMENT URLS (SECTION 5.11.5.2)..... 99

D.17 RETRIEVE LIST OF OUTGOING REQUESTS (SECTION 5.12.3.1) 100

D.18 CREATE OUTGOING MESSAGE, RETURNING THE REPRESENTATION OF CREATED RESOURCE (SECTION 5.12.5.1) 101

D.19 CREATE OUTGOING MESSAGE, RETURNING THE LOCATION OF CREATED RESOURCE (SECTION 5.12.5.2) 102

D.20 CREATE OUTGOING MESSAGE WITH CHARGING (SECTION 5.12.5.3)..... 103

D.21 CREATE OUTGOING MESSAGE, SERVICE EXCEPTION IN CASE OF ADDRESS(ES) FAILURE (SECTION 5.12.5.4) .. 105

D.22 CREATE OUTGOING MESSAGE, MULTIPLE ADDRESSES PARTIAL SUCCESS, WITH DELIVERYINFO LIST IN RESPONSE (SECTION 5.12.5.5) 106

D.23 CREATE OUTGOING MESSAGE, MULTIPLE ADDRESSES PARTIAL SUCCESS, WITHOUT DELIVERYINFO LIST IN RESPONSE (SECTION 5.12.5.6) 108

D.24 CREATE OUTGOING MESSAGE USING SHORT CODE AS SENDERADDRESS, RETURNING THE REPRESENTATION OF CREATED RESOURCE (SECTION 5.12.5.7)..... 109

D.25 READ MESSAGE REQUEST AND DELIVERY STATUS (SECTION 5.13.3.1)..... 111

D.26 READ MESSAGE DELIVERY STATUS (SECTION 5.14.3.1) 112

D.27 READ DELIVERY NOTIFICATION SUBSCRIPTIONS (SECTION 5.15.3.1)..... 112

D.28 CREATE OUTBOUND DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.15.5.1)..... 113

D.29 READ INDIVIDUAL MESSAGE DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.16.3.1)..... 114

D.30 DELETE MESSAGE DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.16.6.1) 114

D.31 NOTIFY CLIENT ABOUT OUTBOUND MESSAGE DELIVERY STATUS, MULTIPLE DELIVERY STATUS PER NOTIFICATION (SECTION 5.17.5.1)..... 114

D.32 NOTIFY CLIENT ABOUT OUTBOUND MESSAGE DELIVERY STATUS, SINGLE DELIVERY STATUS PER NOTIFICATION (SECTION 5.17.5.2)..... 115

APPENDIX E. PARLAY X OPERATIONS MAPPING (INFORMATIVE)..... 116

Figures

Figure 1: Resource structure defined by this specification 13

Figure 2: Send message and check the delivery status	34
Figure 3: Inbound message delivery (push mode)	35
Figure 4: Inbound message delivery (polling mode)	37

Tables

Table 1: Parlay X operations mapping	116
--	-----

1. Scope

This specification defines a RESTful Multi Media 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/>
- [OMA-IM-TS] “Instant Messaging using SIMPLE”, Open Mobile Alliance™, OMA-TS-SIMPLE_IM-V1_0-20100218-D.doc, URL: <http://www.openmobilealliance.org/>
- [OMA_REST_TS_Common] “Common definitions and specifications for OMA REST interfaces”, Open Mobile Alliance™, OMA-TS_REST_Common-V1_0, URL: <http://www.openmobilealliance.org/>
- [REST_TS_Common] “RESTful bindings for Parlay X Web Services – Common”, Open Mobile Alliance™, OMA-TS-ParlayREST_Common-V1, 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>
- [RFC2616] “Hypertext Transfer Protocol -- HTTP/1.1”, R. Fielding et. al, January 1999, URL:<http://www.ietf.org/rfc/rfc2616.txt>
- [RFC4234] “Augmented BNF for Syntax Specifications: ABNF”. D. Crocker, Ed., P. Overell. October 2005, URL:<http://www.ietf.org/rfc/rfc4234.txt>
- [RFC4627] “The application/json Media Type for JavaScript Object Notation (JSON)”, D. Crockford, July 2006, URL:<http://www.ietf.org/rfc/rfc4627.txt>
- [SCRRULES] “SCR Rules and Procedures”, Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:<http://www.openmobilealliance.org/>
- [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>
- [W3C-FORMS] “Use of Forms”. URL:<http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.2>
- [W3C-URLENC] W3C HTML 2.0 Specification, form-urlencoded Media Type, URL: http://www.w3.org/MarkUp/html-spec/html-spec_8.html#SEC8.2.1
- [XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL: <http://www.w3.org/TR/xmlschema-1/>
- [XMLSchema2] W3C Recommendation, XML Schema Part 2: Data types Second Edition, URL: <http://www.w3.org/TR/xmlschema-2/>

2.2 Informative References

- [OMADICT] “Dictionary for OMA Specifications”, Version 2.7, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_7, URL:<http://www.openmobilealliance.org/>
- [REST_WP] “White Paper on Guidelines for ParlayREST API specifications”, Open Mobile Alliance™, OMA-WP-Guidelines-for-ParlayREST-API-specifications, 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].

3.3 Abbreviations

API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
EMS	Enhanced Message Service
HTTP	HyperText Transfer Protocol
IM	Instant Message
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
MMS	Multi Media System
MSISDN	Mobile Subscriber ISDN Number
OMA	Open Mobile Alliance
REST	REpresentational State Transfer
RTX	Ring Tone eXtended
SCR	Static Conformance Requirements
SI	Service Indication
SL	Service Logic
SMS	Short Message Service
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 ParlayREST Technical Specification for Multimedia Messaging contains the HTTP protocol binding for the Parlay X Multimedia Messaging Web Services 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 form-urlencoding).

4.1 Version 1.0

Version 1.0 of the Multimedia Messaging Service ParlayREST API specification supports 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 command)

4.2 Version 1.1

Version 1.1 of the Multimedia Messaging Service ParlayREST API is a maintenance release.

5. Messaging API definition

This section is organized to support a comprehensive understanding of the MMS 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_TS_Common] resp. [OMA_REST_TS_Common].

The remainder of this document is structured as follows:

Section 5 starts with 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.

The remaining subsections in section 5 contain the detailed specification for each of the resources. Each such subsection defines the resource, the request URI variables that are common for all HTTP commands, the possible HTTP response codes, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 5 use XML as the format for the message body. Form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D. Appendix B provides the Static Conformance Requirements (SCR).

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

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 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 [OMA_REST_TS_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.

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

5.1 Resources Summary

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

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.

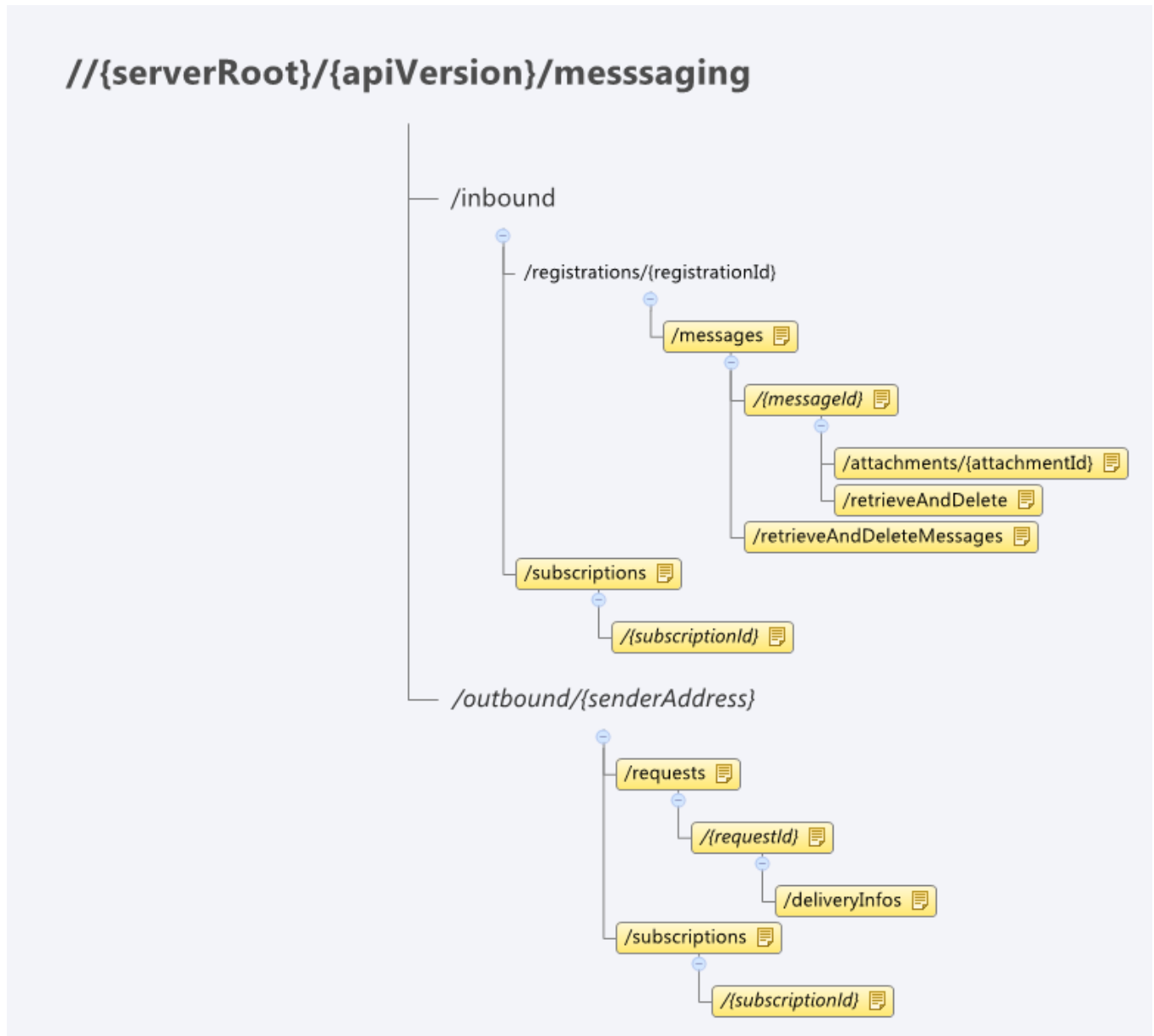


Figure 1: Resource structure defined by this specification

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: Inbound multimedia messages for periodic polling (based on a provisioning step configuration)

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/messaging	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
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 formatted) 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

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/messaging	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
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 attachment from gateway

Purpose: Subscription Management for Inbound messages

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/messaging	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: Callback notifications for Inbound Messages

Resource	URL <specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE

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: Sending message and obtaining the Delivery Status

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/messaging	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: Subscription Management for Outbound Message Delivery Status (overwrites individual request notifications)

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/messaging	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 receipt notification subscription and stop corresponding delivery receipt notifications

Purpose: Callback notifications for 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 Messaging (MMS, SMS, WAP, IM) ParlayREST API Data Structures

The namespace for the Messaging data types is:

urn:oma:xml:rest: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_TS_Common]. The use of the names 'xsd' and 'common' is not semantically significant.

5.2.1 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 request and/or response bodies.

5.2.2 Type: InboundMessage

Individual inbound message.

Element	Type	Optional	Description
destinationAddress	xsd:anyURI	No	Number associated with the invoked Message service, i.e. the destination address used by the terminal to send the message.
senderAddress	xsd:anyURI	No	Indicates message senderAddress.
dateTime	xsd:dateTime	Yes	Time when message was received by operator
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
messageId	xsd:string	Yes	OPTIONAL 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.
inboundSMSTextMessage	InboundSMSTextMessage	Choice	Inbound SMS Text Message
inboundMMSMessage	InboundMMSMessage	Choice	Inbound MMS Message
inboundIMMessage	InboundIMMessage	Choice	Inbound IM Message

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

5.2.3 Type: InboundMessageNotification

Notification about an individual inbound message.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	CallbackData as passed by the application during the associated Send MMS operation. See [REST_TS_Common] for details.
inboundMessage	InboundMessage	No	Multimedia message
link	common:Link[0..unbounded]	Yes	Link to other resources. For example we can have 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.

5.2.4 Type: InboundSMSTextMessage

Content of an inbound textual SMS message.

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

5.2.5 Type: InboundMMSMessage

Parameters for an inbound 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.
link	common:Link[0..unbounded]	Yes	<p>Link to other resources (like individual attachments: <link rel="attachment" href="../inbound/registration/{registrationId}/messages/{messageId}/attachments/{attachmentId}"/>)</p> <p>In case the message contains a presentation part, this SHALL be referenced by the first item in the list of <link> elements.</p>
bodyText	xsd:string	Yes	Contains the message body if it is encoded as ASCII text

5.2.6 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.
link	common:Link[0..unbounded]	Yes	<p>Link to other resources (like individual attachments: <link rel="attachment" href="../inbound/registration/{registrationId}</p>

			<p>/messages/{messageId}/attachments/{attachmentId}"/></p> <p>In case the message contains a presentation part, this SHALL be referenced by the first item in the list of <link> elements.</p>
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

5.2.7 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 request and/or response bodies

5.2.8 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 multimedia message
criteria	xsd:string	Yes	<p>The text to match against to determine the application to receive the notification.</p> <p>This text is matched against the first word, as 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.</p> <p>If the subject of the multimedia</p>

			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 TS 23.140]
clientCorrelator	xsd:string	Yes	A correlator that the client MAY use to tag this particular resource representation during a request to create a resource on the server. 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
useAttachmentURLs	xsd:boolean	Yes	Default: false If set to 'true', inbound message has links to attachments. 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_TS_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_TS_Common] provides a recommendation regarding the generation of the value of this field.

5.2.9 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. Otherwise, inbound message includes attachments using MIME

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

5.2.10 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 request and/or response bodies.

5.2.11 Type: OutboundMessageRequest

Individual outbound message request.

Element	Type	Optional	Description
address	xsd:anyURI [1..unbounded]	No	Destination addresses for the Message.
senderAddress	xsd:anyURI	No	The address of the sender to whom a responding message may be sent. If the address is in the form of an MSISDN, include the protocol prefix 'tel' and '%3A%2B' followed by the country code before the subscriber number; e.g. tel%3A%2B19585550100. 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).
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.
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.
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
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 re-sending the message 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, outboundSMSLogoMessage, outboundSMSRingToneMessage, outboundWAPMessage, outboundMMSMessage or outboundIMMessage.

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 (POL0001: Policy error SHALL be used in case the message is rejected, see [REST_TS_Common]).

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

Regarding the clientCorrelator field, the note in section 5.2.8 applies.

5.2.12 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.

5.2.13 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	<p>There is no user intervention.</p> <p>If the parameter is not specified, the default value will be "ExecuteLow". See [WAP-SL] for more details.</p> <p>May be present only if ContentType is "ServiceLoading"</p>
serviceIndicationAction	ServiceIndicationAction	Choice	<p>Allows controlling the level of intrusiveness, of outbound wap push messages.</p> <p>According to [WAP-SI] it contains a text string specifying the action to be taken when the message is received.</p> <p>If the parameter is not specified, the value "SignalMedium" is used</p> <p>May be present only if ContentType is "ServiceIndication"</p>
text	xsd:string	Yes	<p>Information that accompanies the push.</p> <p>May be present only if ServiceIndicationAction is present and ContentType is "ServiceIndication"</p>
created	xsd:dateTime	Yes	<p>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.</p> <p>May be present only if ContentType is "ServiceIndication".</p>

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

5.2.14 Type: OutboundSMSTextMessage

Content of an outbound textual SMS message.

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

5.2.15 Type: OutboundSMSBinaryMessage

Content of an outbound binary SMS message.

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

5.2.16 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'

5.2.17 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'

5.2.18 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

5.2.19 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 request and/or response bodies.

5.2.20 Type: DeliveryInfoNotification

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

Element	Type	Optional	Description
callbackData	xsd:string	Yes	CallbackData if passed by the application in the receiptRequest element during the associated Send message operation. See [REST_TS_Common] for details.
deliveryInfo	DeliveryInfo [1..unbounded]	No	Delivery Information
link	common:Link[0..unbounded]	Yes	Link to other resources. For example we can have a link to the original outbound message request.

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

5.2.21 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
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. For example we can have a link to the original outbound message request.

5.2.22 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 request and/or response bodies.

5.2.23 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 MAY use to tag this particular resource representation during a request to create a resource on the server. 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

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

Regarding the clientCorrelator field, the note in section 5.2.8 applies.

5.2.24 Enumeration: DeliveryStatus

Delivery status enumeration.

Enumeration	Description
DeliveredToTerminal	Successful delivery to 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 multimedia 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 SendMessage is not supported.

5.2.25 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 with multimedia, as defined in [OMA-IM-TS].
IMFileTransfer	Large IM used for File Transfer, as defined in [OMA-IM-TS]

5.2.26 Enumeration: MessagePriority

Message priority enumeration.

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

5.2.27 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

5.2.28 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.

5.2.29 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 a 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.

5.2.30 Enumeration: SmsFormat

SMS format enumeration.

Enumeration	Description
Ems	EMS conversion
SmartMessaging	SmartMessaging conversion

5.2.31 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]

5.2.32 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
- 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.

5.2.33 MIME multipart representation

MMS messages (inbound and 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_TS_Common]. The type of the form entry carrying the root fields 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

5.3.1 Send message and check the delivery status

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

The resources:

- To send a multimedia message, create new resource under
http://{serverRoot}/{apiVersion}/messaging/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}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
 - b. directly read the resource
http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/deliveryInfos

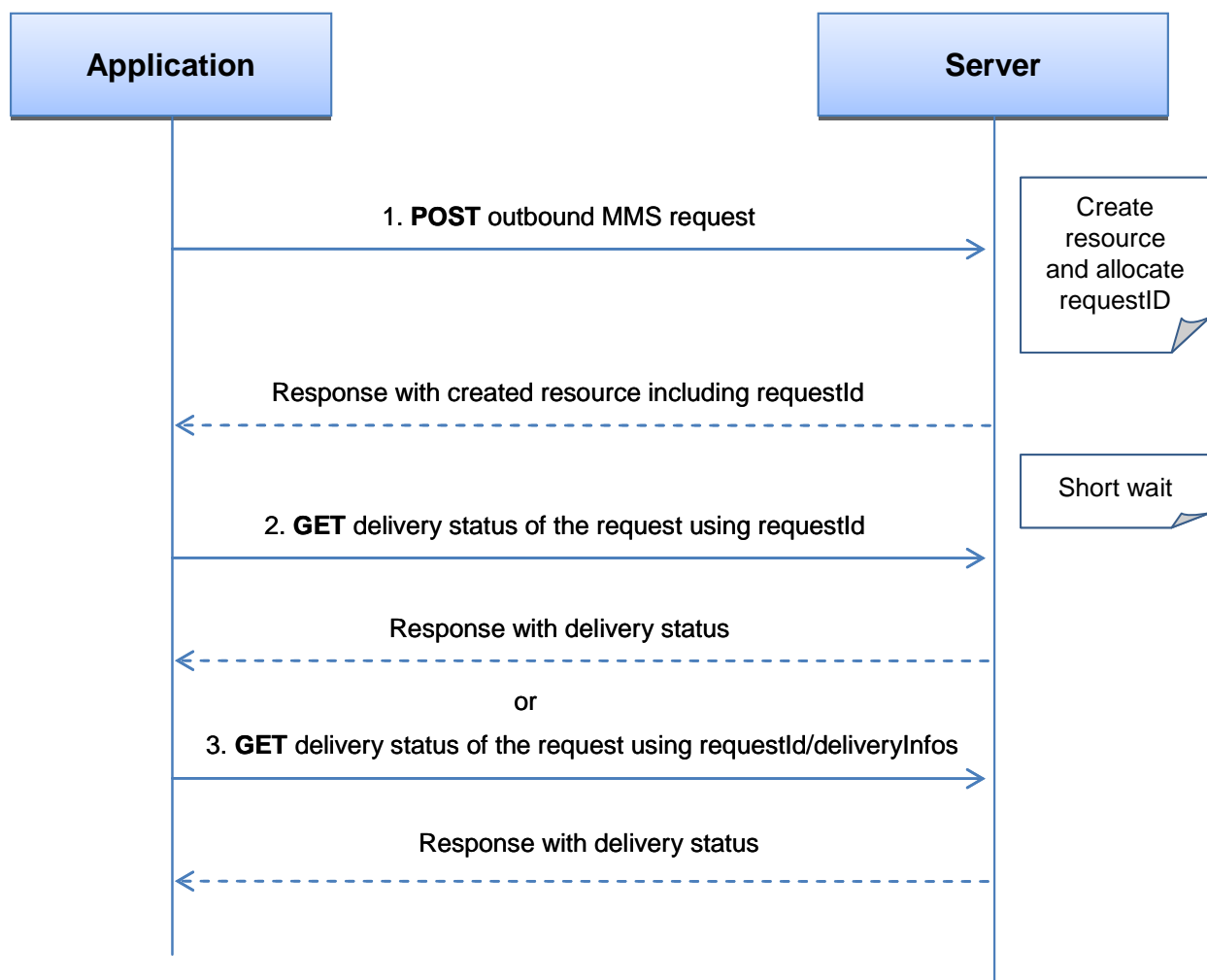


Figure 2: Send message and check the delivery status

Outline of the flows:

1. An application initiates the creation of new outbound message request using POST and receives the created request 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 resources:

- To start subscription to notifications for inbound messages, create new resource under **http://{serverRoot} /{apiVersion}/messaging/inbound/subscriptions**
- To notify the application about the message arrival, POST notification to the client supplied notifyURL
- To stop the subscription to notifications, delete the resource **http://{serverRoot} /{apiVersion}/messaging/inbound/subscriptions/{subscriptionId}**

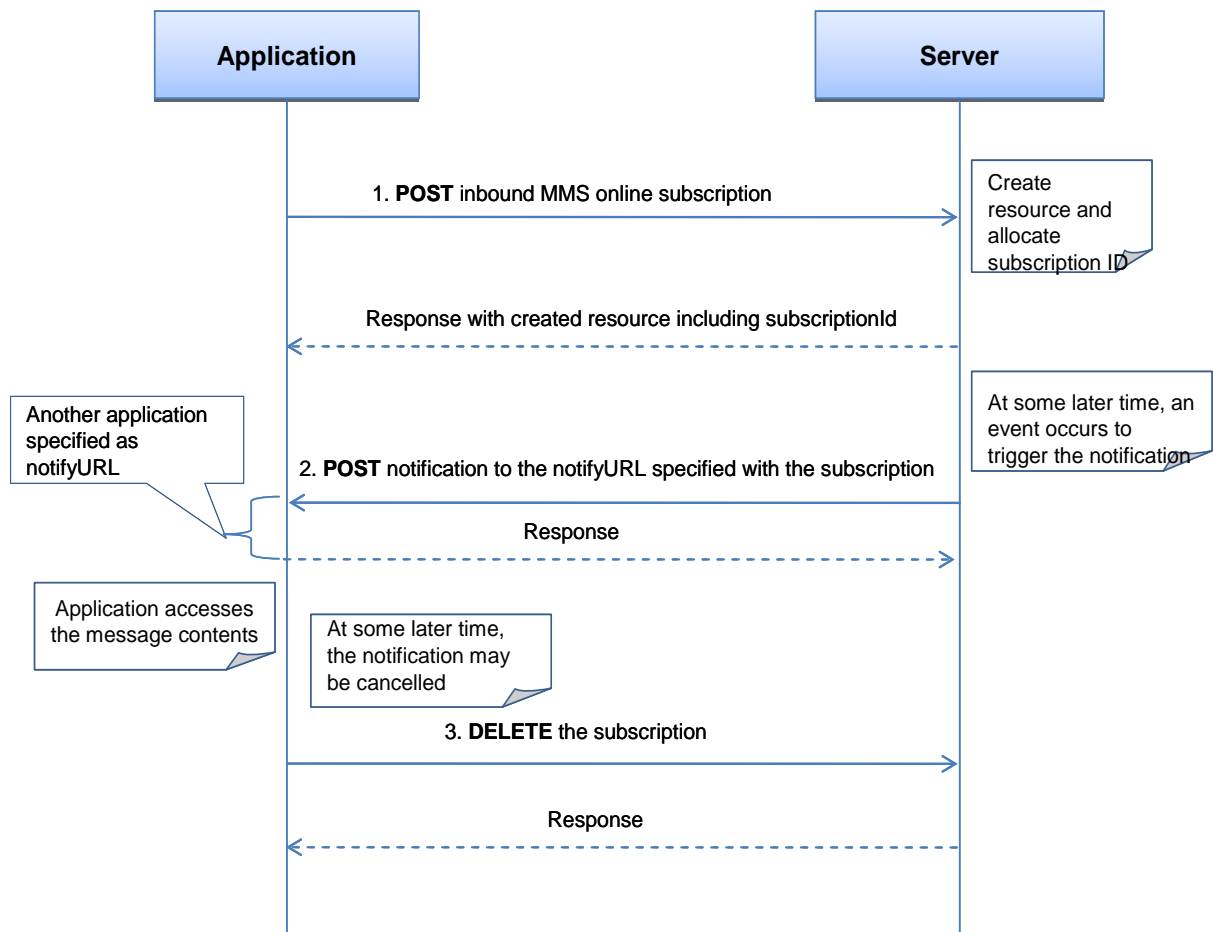


Figure 3: Inbound message delivery (push mode)

Outline of the flows:

1. An application subscribes to notifications for inbound messages using POST 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.
3. 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} /{apiVersion}/messaging/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} /{apiVersion}/messaging/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 remove one message from the storage, delete the resource
http://{serverRoot} /{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}

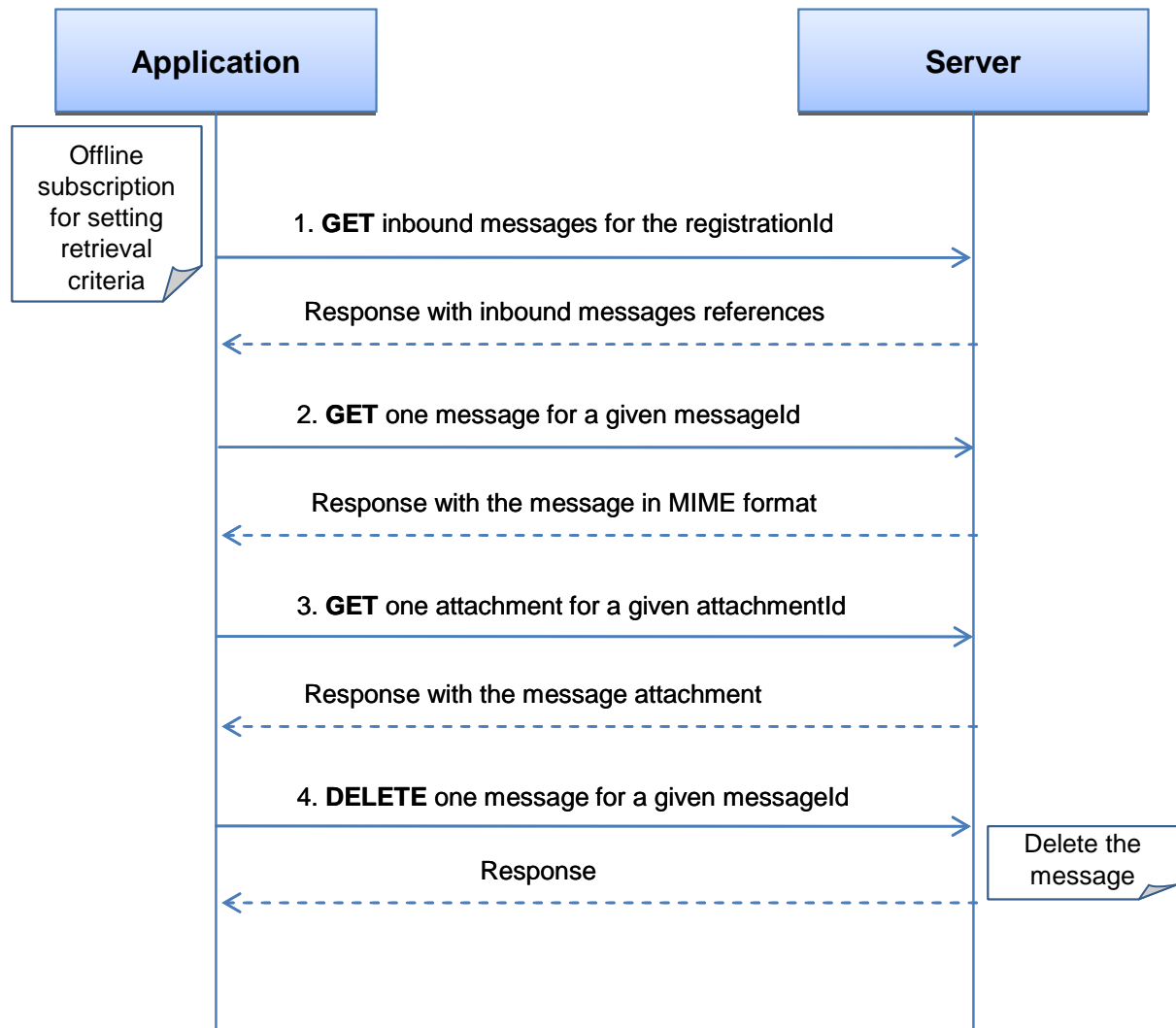


Figure 4: Inbound message delivery (polling mode)

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 request using GET with a resource URL containing the messageId
3. The application reads one attachment to the message using GET with a resource URL containing the attachmentId.
4. The application removes one of the messages from gateway storage using DELETE with a resource URL containing the messageId .

5.4 Resource: Inbound messages for a given registration

The resource used is:

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

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

5.4.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API clients wants to use (e.g. 1 for version 1.x)
registrationId	reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier

5.4.2 Response Codes

5.4.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.4.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.4.3 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 command (see DELETE on Inbound message).

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 is there are more then one pending
useAttachmentURLs	xsd:boolean	Yes	Default: false If set to 'true', inbound message would have links to attachments. 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.

5.4.3.1 Examples 1: Retrieve messages for a registration, useAttachmentURLs=false

(Informative)

5.4.3.1.1 Request

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.4.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageList xmlns:mms="urn:oma:xml:rest:messaging:1">
  <!-- MMS -->
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/1/messaging/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/1/messaging/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/1/messaging/inbound/registrations/reg123/
messages?maxBatchSize=2</resourceURL>
</mms:inboundMessageList>
```

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

(Informative)

5.4.3.2.1 Request

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.4.3.2.2 Response

```

HTTP/1.1 404 Not Found
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:common:1">
  <link rel="self" href="http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2" />
  <serviceException>
    <messageId>SVC0002</messageId>
    <text>Invalid input value. The requested registration id: %1 does not exist </text>
    <variables>reg123</variables>
  </serviceException>
</common:requestError>

```

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

5.4.3.3.1 Request

```

GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2&useAttachmentURLs=true HTTP/1.1
Accept: application/xml
Host: example.com:80

```

5.4.3.3.2 Response

```

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageList xmlns:mms="urn:oma:xml:rest:messaging:1">
  <!-- MMS -->
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123</resourceURL>
    <messageId>msg123</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
      <link rel="attachment"
        href="http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/
        attachments/attach123" />
      <link rel="attachment"
        href="http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/
        attachments/attach124" />
      <bodyText>See attached picture</bodyText>
    </inboundMMSMessage>
  </inboundMessage>
  <totalNumberOfPendingMessages>20</totalNumberOfPendingMessages>
  <numberOfMessagesInThisBatch>2</numberOfMessagesInThisBatch>
</mms:inboundMessageList>
<resourceURL>http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/

```



```
msg123/attachments</resourceURL>
</mms:inboundMessageList>
```

5.4.4 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 section 14.7 of [RFC 2616].

5.4.5 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 section 14.7 of [RFC 2616].

5.4.6 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 section 14.7 of [RFC 2616].

5.5 Resource: Inbound messages retrieve and delete using registration

The resource used is:

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

This resource is used for retrieving and deleting the list of incoming messages using retrieval criteria that is setup 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.

5.5.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API clients wants to use (e.g. 1 for version 1.x)
registrationId	reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier

5.5.2 Response Codes

5.5.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.5.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.5.3 GET

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

5.5.4 PUT

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

5.5.5 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 URL provided in mime-url attribute.

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

5.5.5.1 Example: Retrieve and delete inbound messages (Informative)

5.5.5.1.1 Request

```
POST /exampleAPI/1/messaging/inbound/registrations/reg123/messages/retrieveAndDeleteMessages HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

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

5.5.5.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageList xmlns:mms="urn:oma:xml:rest: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/1/messaging/inbound/registrations/reg123/retrieveAndDeleteMessages</resourceURL>
</mms:inboundMessageList>

```

5.5.6 DELETE

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

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

The resource used is:

http://{serverRoot}/{apiVersion}/messaging/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}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}** resource should be used for reliable delivery.

5.6.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API clients wants to use (e.g. 1 for version 1.x)
registrationId	reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier

messageld	unique message identifier generated by server
-----------	---

5.6.2 Response Codes

5.6.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.6.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.6.3 GET

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

5.6.4 PUT

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

5.6.5 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.

5.6.5.1 Example: Read and delete one message

(informative)

5.6.5.1.1 Request

```
POST /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/retrieveAndDelete HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
```

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

5.6.5.1.2 Response

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

```

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

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

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

Look at the attached picture

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

GIF89a...binary image data...

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

```

5.6.6 DELETE

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

5.7 Resource: Inbound message for a given registration

The resource used is:

http://{serverRoot}/{apiVersion}/messaging/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.

5.7.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API clients wants to use (e.g. 1 for version 1.x)
registrationId	reference to the provisioned in advance and known to the client application. Analogous to Parlay X registrationIdentifier
messageId	unique message identifier generated by server

5.7.2 Response Codes

5.7.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.7.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.7.3 GET

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

5.7.3.1 Example: Read message from gateway storage (Informative)

5.7.3.1.1 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 [OMA_REST_TS_Common].

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123?resFormat=XML HTTP/1.1
Host: example.com:80
```

5.7.3.1.2 Response

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

```

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

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

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

Look at the attached picture

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

GIF89a...binary image data...

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

```

5.7.4 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 section 14.7 of [RFC 2616].

5.7.5 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 section 14.7 of [RFC 2616].

Note: See Inbound MMS message retrieve and delete.

5.7.6 DELETE

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

5.7.6.1 Example: Remove message from gateway storage (Informative)

5.7.6.1.1 Request

```
DELETE /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.7.6.1.2 Response

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

5.8 Resource: Inbound message attachment

The resource used is:

http://{serverRoot}/{apiVersion}/messaging/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.

5.8.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
registrationId	reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier
messageId	unique message identifier generated by server
attachmentId	unique attachment identifier generated by server

5.8.2 Response Codes

5.8.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.8.3 GET

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

5.8.3.1 Example: Read an MMS attachment (Informative)

5.8.3.1.1 Request

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: image/gif, image/png, image/jpeg, text/html, application/xml
Host: example.com:80
```

5.8.3.1.2 Response

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

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

5.8.4 PUT

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

5.8.5 POST

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

5.8.6 DELETE

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

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

5.8.6.1.1 Request

```
DELETE /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/attachments/attach123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.8.6.1.2 Response

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

5.9 Resource: Inbound message subscriptions

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

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

5.9.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API clients want to use (e.g. version 1 for 1.x)

5.9.2 Response Codes

5.9.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.9.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.9.3 GET

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

5.9.3.1 Example: Read active subscriptions (Informative)

5.9.3.1.1 Request

```
GET /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.9.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
```

Date: Thu, 04 Jun 2009 02:51:59 GMT

```
<?xml version="1.0" encoding="UTF-8"?>
<mms:subscriptionList xmlns:mms="urn:oma:xml:rest: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/1/messaging/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/1/messaging/inbound/subscriptions/0000002</resourceURL>
    <useAttachmentURLs>false</useAttachmentURLs>
  </subscription>
</mms:subscriptionList>
```

5.9.4 PUT

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

5.9.5 POST

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

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

5.9.5.1.1 Request

```
POST /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
```

```
<?xml version="1.0" encoding="UTF-8"?>
<mms:subscription xmlns:mms="urn:oma:xml:rest: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>
  <useAttachmentURLs>>false</useAttachmentURLs>
</mms:subscription>

```

5.9.5.1.2 Response

```

HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

```

<?xml version="1.0" encoding="UTF-8"?>
<mms:subscription xmlns:mms="urn:oma:xml:rest: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/1/messaging/inbound/subscriptions/sub123</resourceURL>
  <useAttachmentURLs>>false</useAttachmentURLs>
</mms:subscription>

```

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

5.9.5.2.1 Request

```

POST /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

```

```

<?xml version="1.0" encoding="UTF-8"?>
<mms:subscription xmlns:mms="urn:oma:xml:rest: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>
  <useAttachmentURLs>>false</useAttachmentURLs>
</mms:subscription>

```

5.9.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:common:1">
  <resourceURL>http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123</resourceURL>
</common:resourceReference>
```

5.9.6 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 section 14.7 of [RFC 2616].

5.10 Resource: Individual inbound message subscription

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

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

5.10.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
subscriptionId	identifies the subscription

5.10.2 Response Codes

5.10.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.10.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.10.3 GET

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

5.10.3.1 Example: Read individual subscription

(Informative)

5.10.3.1.1 Request

```
GET /exampleAPI/1/messaging/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.10.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:subscription xmlns:mms="urn:oma:xml:rest: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/1/messaging/inbound/subscriptions/sub123</resourceURL>
  <useAttachmentURLs>>false</useAttachmentURLs>
</mms:subscription>
```

5.10.4 PUT

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

5.10.5 POST

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

5.10.6 DELETE

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

5.10.6.1 Example: Delete a subscription

(Informative)

5.10.6.1.1 Request

```
DELETE /exampleAPI/1/messaging/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.10.6.1.2 Response

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

5.11 Resource: Client notification about inbound message

This resource is a client provided callback URL for posting notifications about incoming messages. ParlayREST does not make any assumption about the structure of this URL.

5.11.1 Request URI variables

Client provided.

5.11.2 Response Codes

5.11.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.11.3 GET

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

5.11.4 PUT

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

5.11.5 POST

This operation is used to notify client about message arrival.

5.11.5.1 Example 1: Message arrival notification

(Informative)

5.11.5.1.1 Request

```
POST /application/example.com/notifications/DeliveryInfoNotification/88888 HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageNotification xmlns:mms="urn:oma:xml:rest:messaging:1">
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123
    </resourceURL>
    <link rel="Subscription" href="http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123"/>
  </inboundMessage>
</mms:inboundMessageNotification>
```

```

<messageId>msg123</messageId>
<inboundMMSMessage>
  <subject>Who is RESTing on the beach?</subject>
</inboundMMSMessage>
</inboundMessage>
</mms:inboundMessageNotification>

```

5.11.5.1.2 Response

```

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

```

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

5.11.5.2.1 Request

```

POST /application/example.com/notifications/DeliveryInfoNotification/88888 HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageNotification xmlns:mms="urn:oma:xml:rest:messaging:1">
  <inboundMessage>
    <destinationAddress>tel:+19585550100</destinationAddress>
    <senderAddress>tel:+19585550101</senderAddress>
    <resourceURL>http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123
    </resourceURL>
    <link rel="Subscription" href="http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123"/>
    <messageId>msg123</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
      <link rel="attachment"
        href="http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/
        attachments/attach123" />
      <link rel="attachment"
        href="http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/
        attachments/attach124" />
      <bodyText>Look at the attached picture</bodyText>
    </inboundMMSMessage>
  </inboundMessage>
</mms:inboundMessageNotification>

```

5.11.5.2.2 Response

```

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

```


5.11.6 DELETE

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

5.12 Resource: Outbound message requests

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

This resource is used for sending outbound messages.

5.12.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: <code>http://example.com/exampleAPI</code>
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
senderAddress	sender application address. Typically SHORT CODE [OMA_REST_TS_Common], but could be a terminal address

5.12.2 Response Codes

5.12.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.12.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.12.3 GET

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

5.12.3.1 Example: Retrieve list of outgoing requests (Informative)

5.12.3.1.1 Request

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.12.3.1.2 Response

```
HTTP/1.1 200 OK
```

```

Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequestList xmlns:mms="urn:oma:xml:rest: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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
    <deliveryInfoList>
      <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos
      </resourceURL>
    <deliveryInfo>
      <address>tel:+19585550103</address>
      <deliveryStatus>DeliveredToTerminal</deliveryStatus>
    </deliveryInfo>
  </deliveryInfoList>
</outboundMessageRequest>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests</resourceURL>
</mms:outboundMessageRequestList>

```

5.12.4 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 section 14.7 of [RFC 2616].

5.12.5 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.

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

5.12.5.1.1 Request

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

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

```

```

<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>

```

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

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

```
====12345====
```

```
Content-Disposition: attachments; 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-----
```

5.12.5.1.2 Response

```

HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

```

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>

```

```

</receiptRequest>
<outboundMMSMessage>
  <subject>hello from the rest of us!</subject>
  <priority>High</priority>
</outboundMMSMessage>
<clientCorrelator>567895</clientCorrelator>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</mms:outboundMessageRequest>

```

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

5.12.5.2.1 Request

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>

====123456==

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

====12345===
Content-Disposition: attachments; 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====

```

5.12.5.2.2 Response

```

HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

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

```

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

5.12.5.3.1 Request

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="=====123456===";
Host: example.com:80
MIME-Version: 1.0
Host: example.com:80
Accept: application/xml
Content-Length: nnnn

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>

```

```

<outboundMMSMessage>
  <subject>hello from the rest of us!</subject>
  <priority>High</priority>
</outboundMMSMessage>
<clientCorrelator>567896</clientCorrelator>
</mms:outboundMessageRequest>

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

====12345====
Content-Disposition: attachments; 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----

```

5.12.5.3.2 Response for charging not supported

```

HTTP/1.1 400 Bad request
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:common:1">
  <policyException>
    <messageId>POL0008</messageId>
    <text>Charging is not supported</text>
  </policyException>
</common:requestError>

```

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

5.12.5.4.1 Request

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456====";
Content-Length: nnnn
Accept: application/xml

```

```

Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>

-----123456==

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

====12345====
Content-Disposition: attachments; 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====

```

5.12.5.4.2 Response

```

HTTP/1.1 400 Bad Request
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:common:1">
  <serviceException>

```

```

<messageId>SVC0001</messageId>
<text>A service error occurred. Error code is %1 </text>
<variables>ERROR-XYZ</variables>
</serviceException>
</common:requestError>

```

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

5.12.5.5.1 Request

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>

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

====12345===
Content-Disposition: attachments; 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=====
```

5.12.5.5.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
  <deliveryInfoList>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos
</resourceURL>
  <deliveryInfo>
    <address>tel:+19585550103</address>
    <deliveryStatus>MessageWaiting</deliveryStatus>
  </deliveryInfo>
  <deliveryInfo>
    <address>tel:+19585550104</address>
    <deliveryStatus>DeliveryImpossible</deliveryStatus>
  </deliveryInfo>
</deliveryInfoList>
</mms:outboundMessageRequest>
```

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

5.12.5.6.1 Request

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="=====123456====";
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
```

```

MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>

-----123456==

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

====12345====
Content-Disposition: attachments; 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----

```

5.12.5.6.2 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
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123

```

Date: Thu, 04 Jun 2009 02:51:59 GMT

```
<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
</mms:outboundMessageRequest>
```

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

5.12.5.7.1 Request

```
POST /exampleAPI/1/messaging/outbound/72654/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/xml
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0
```

```
====123456==
```

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

```
<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:+19585550103</address>
  <address>tel:+19585550104</address>
  <senderAddress>72654</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/99999</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</mms:outboundMessageRequest>
```

```
====123456==
```

```
Content-Disposition: multipart/form-data; name="attachments"
```

```
Content-Type: multipart/mixed; boundary="====12345===="
```

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

```
Content-Disposition: attachments; 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-----
```

5.12.5.7.2 Response

```
HTTP/1.1 201 Created
```

```
Content-Type: application/xml
```

```
Content-Length: nnnn
```

```
Location: http://example.com/exampleAPI/1/messaging/outbound/72654/requests/req123
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

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

```
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
```

```
<address>tel:+19585550103</address>
```

```
<address>tel:+19585550104</address>
```

```
<senderAddress>72654</senderAddress>
```

```
<senderName>MyName</senderName>
```

```
<outboundMMSMessage>
```

```
<subject>Holiday greetings</subject>
```

```
</outboundMMSMessage>
```

```
<clientCorrelator>567895</clientCorrelator>
```

```
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/72654/requests/req123</resourceURL>
```

```
</mms:outboundMessageRequest>
```

5.12.6 DELETE

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

5.13 Resource: Outbound message request and delivery status

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

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

5.13.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
senderAddress	identifies client application. Typically SHORT CODE [OMA_REST_TS_Common], but could be a terminal address
requestId	outbound message request identifier generated by server

5.13.2 Response Codes

5.13.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.13.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.13.3 GET

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

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

5.13.3.1.1 Request

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.13.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest: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/99999</notifyURL>
  <callbackData>12345</callbackData>
</receiptRequest>
<outboundMMSMessage>
  <subject>Holiday greetings</subject>
</outboundMMSMessage>
<clientCorrelator>567895</clientCorrelator>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123</resourceURL>
<deliveryInfoList>
  <!-- this is optional -->
  <resourceURL>http://example.com/exampleAPI/1/messaging/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>
</mms:outboundMessageRequest>

```

5.13.4 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 section 14.7 of [RFC 2616].

5.13.5 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 section 14.7 of [RFC 2616].

5.13.6 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 section 14.7 of [RFC 2616].

5.14 Resource: Outbound message delivery status

The resource used is:

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

This resource is used to request outbound message delivery status.

5.14.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example:

	http://example.com/exampleAPI
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
senderAddress	identifies client application. Typically SHORT CODE [OMA_REST_TS_Common], but could be a terminal address
requestId	outbound message request identifier generated by server

5.14.2 Response Codes

5.14.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.14.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.14.3 GET

This operation is used to retrieve outgoing message delivery status.

5.14.3.1 Example: Read message delivery status (Informative)

5.14.3.1.1 Request

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.14.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryInfoList xmlns:mms="urn:oma:xml:rest:messaging:1">
  <resourceURL>http://example.com/exampleAPI/1/messaging/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>
</mms:deliveryInfoList>
```

5.14.4 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 section 14.7 of [RFC 2616].

5.14.5 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 section 14.7 of [RFC 2616].

5.14.6 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 section 14.7 of [RFC 2616].

5.15 Resource: Outbound message delivery notification subscriptions

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

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

5.15.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: <code>http://example.com/exampleAPI</code>
apiVersion	version of the ParlayREST API clients wants to use (e.g. 1 for version 1.x)
senderAddress	identifies client application. Typically SHORT CODE [OMA_REST_TS_Common], but could be a terminal address.

5.15.2 Response Codes

5.15.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.15.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.15.3 GET

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

5.15.3.1 Example: Read delivery notification subscriptions (Informative)

5.15.3.1.1 Request

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.15.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryReceiptSubscriptionList xmlns:mms="urn:oma:xml:rest:messaging:1">
  <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/subscriptions</resourceURL>
  <deliveryReceiptSubscription>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/99999</notifyURL>
      <callbackData>12345</callbackData>
    </callbackReference>
    <filterCriteria>0102</filterCriteria>
    <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123</resourceURL>
  </deliveryReceiptSubscription>
  <deliveryReceiptSubscription>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
      <callbackData>54321</callbackData>
    </callbackReference>
    <filterCriteria>0103</filterCriteria>
    <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123</resourceURL>
  </deliveryReceiptSubscription>
</mms:deliveryReceiptSubscriptionList>
```

5.15.4 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 section 14.7 of [RFC 2616].

5.15.5 POST

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

5.15.5.1 Example: Create outbound delivery notification subscription (Informative)

5.15.5.1.1 Request

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
```

```

Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

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

```

5.15.5.1.2 Response

```

HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryReceiptSubscription xmlns:mms="urn:oma:xml:rest:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
  <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123</resourceURL>
</mms: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 5.9.5.2.2).

5.15.6 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 section 14.7 of [RFC 2616].

5.16 Resource: Individual outbound message delivery notification subscription

The resource used is:

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

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

5.16.1 Request URI variables

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

Name	Description
serverRoot	server base url: hostname+port+base path. Example: http://example.com/exampleAPI
apiVersion	version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)
senderAddress	sender application address. Typically SHORT CODE [OMA_REST_TS_Common], but could be a terminal address.
subscriptionId	identifier of the subscription.

5.16.2 Response Codes

5.16.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.16.2.2 Exception fault codes

For Policy Exception and Service Exception fault codes applicable to Multimedia Messaging, see [3GPP 29.199-5].

5.16.3 GET

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

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

5.16.3.1.1 Request

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.16.3.1.2 Response

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryReceiptSubscription xmlns:mms="urn:oma:xml:rest:messaging:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/77777</notifyURL>
    <callbackData>12345</callbackData>
```

```
</callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123</resourceURL>
</mms:deliveryReceiptSubscription>
```

5.16.4 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 section 14.7 of [RFC 2616].

5.16.5 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 section 14.7 of [RFC 2616].

5.16.6 DELETE

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

5.16.6.1 Example: Delete message delivery notification subscription (Informative)

5.16.6.1.1 Request

```
DELETE /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions /sub123 HTTP/1.1
Accept: application/xml
Host: example.com:80
```

5.16.6.1.2 Response

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

5.17 Resource: Client notification about outbound message delivery status

This resource is a client provided callback URL for client notification about outbound message delivery status. ParlayREST does not make any assumption about the structure of this URL.

5.17.1 Request URI variables

Client provided.

5.17.2 Response Codes

5.17.2.1 HTTP Response Codes

For HTTP response codes, see [OMA_REST_TS_Common].

5.17.3 GET

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

5.17.4 PUT

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

5.17.5 POST

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

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

5.17.5.1.1 Request

```
POST /application.example.com/notifications/DeliveryInfoNotification/77777 HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryInfoNotification xmlns:mms="urn:oma:xml:rest: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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123" />
</mms:deliveryInfoNotification>
```

5.17.5.1.2 Response

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

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

5.17.5.2.1 Request

```
POST /application.example.com/notifications/DeliveryInfoNotification/77777 HTTP/1.1
```

```
Content-Type: application/xml
Accept: application/xml
Content-Length: nnnn
Host: application.example.com:80

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

5.17.5.2.2 Response

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

5.17.6 DELETE

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

Appendix A. Change History

(Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-TS-ParlayREST_MultiMediaMessaging-V1_1-20120724-A	24 Jul 2012	Status changed to Approved by TP Ref TP Doc# OMA-TP-2012-0280-INP_ParlayREST_2_0_for_Final_Approval

Appendix B. Static Conformance Requirements (Normative)

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

B.1 SCR for ParlayREST.MMS Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-SUPPORT-S-001-M	Support for the MMS REST API	5	
PARLAYREST-MMS-SUPPORT-S-002-M	Support for the XML request & response format	5	
PARLAYREST-MMS-SUPPORT-S-003-M	Support for the JSON request & response format	5	
PARLAYREST-MMS-SUPPORT-S-004-O	Support for the application/form-urlencoded format	Appendix C	

B.1.1 SCR for ParlayREST.MMS.Inbound.Registration Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-OFF-S-001-M	Support for reliable inbound messages delivery	5.4	
PARLAYREST-MMS-INB-OFF-S-002-M	Retrieve messages from server - GET	5.4.3	

B.1.2 SCR for ParlayREST.MMS.Inbound.Registration.RetrieveDelete Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-OFF-RETDEL-S-001-O	Support for inbound message delivery and delete	5.5	PARLAYREST-MMS-INB-OFF-RETDEL-S-002-O
PARLAYREST-MMS-INB-OFF-RETDEL-S-002-O	Retrieve and delete messages from server - POST	5.5.5	

B.1.3 SCR for ParlayREST.MMS.Individual.Inbound.Registration.RetrieveDelete Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-MIME-INB-OFF-RETDEL-S-001-O	Support for inbound message delivery and delete	5.6	PARLAYREST-MMS-MIME-INB-OFF-RETDEL-S-002-O
PARLAYREST-MMS-MIME-INB-OFF-RETDEL-S-002-O	Retrieve and delete one message from server - POST	5.6.5	

B.1.4 SCR for ParlayREST.MMS.Individual.Inbound Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-IND-INB-S-001-M	Support for inbound individual message delivery	5.7	
PARLAYREST-MMS-IND-INB-S-002-O	Retrieve one message from server - GET	5.7.3	
PARLAYREST-MMS-IND-INB-S-003-M	Confirm and delete retrieved message from server - DELETE	5.7.6	

B.1.5 SCR for ParlayREST.MMS.Attach.Individual.Inbound Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-ATTACH-IND-INB-S-001-O	Support for inbound individual message attachment delivery	5.8	PARLAYREST-MMS-ATTACH-IND-INB-S-002-O AND PARLAYREST-MMS-ATTACH-IND-INB-S-003-O
PARLAYREST-MMS-ATTACH-IND-INB-S-002-O	Retrieve one message attachment from server - GET	5.8.3	
PARLAYREST-MMS-ATTACH-IND-INB-S-003-O	Confirm and delete retrieved message attachment from server - DELETE	5.8.6	

B.1.6 SCR for ParlayREST.MMS.Inbound.Subscr Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-ONL-SUBSCR-S-001-M	Support inbound subscriptions	5.9	
PARLAYREST-MMS-INB-ONL-SUBSCR-S-002-O	Read active subscriptions - GET	5.9.3	
PARLAYREST-MMS-INB-ONL-SUBSCR-S-003-M	Create inbound message subscription – POST (XML and JSON)	5.9.5	
PARLAYREST-MMS-INB-ONL-SUBSCR-S-004-O	Create inbound message subscription – POST (www-form-urlencoded)	C.3	

B.1.7 SCR for ParlayREST.MMS.Inbound.Individual.Subscr Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-INDON-SUBSCR-S-001-M	Support for control and read access to individual inbound subscription	5.10	

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-INDON-SUBSCR-S-002-O	Read individual inbound subscription - GET	5.10.3	
PARLAYREST-MMS-INB-INDON-SUBSCR-S-003-M	Update individual inbound subscriptions - DELETE	5.10.6	

B.1.8 SCR for ParlayREST.MMS.Inbound.Notifications Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-INB-NOTIF-S-001-M	Support for notifying application about inbound messages	5.11	
PARLAYREST-MMS-INB-NOTIF-S-002-M	Notify application about inbound message arrival - POST	5.11.5	

B.1.9 SCR for ParlayREST.MMS.Outbound Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-OUTB-S-001-M	Support for outbound messages	5.12	
PARLAYREST-MMS-OUTB-S-002-O	Retrieve list of pending outgoing message requests - GET	5.12.3	
PARLAYREST-MMS-OUTB-S-003-M	Create outgoing message request - POST (XML and JSON)	5.12.5	
PARLAYREST-MMS-OUTB-S-004-O	Create outgoing message request - POST (www-form-urlencoded)	C.1	

B.1.10 SCR for ParlayREST.MMS.Outbound.MsgAndDeliveryStatus Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-OUTB-MSGDELSTAT-S-001-O	Support for requesting an outbound message and its delivery status	5.13	PARLAYREST-MMS-OUTB-MSGDELSTAT-S-002-O
PARLAYREST-MMS-OUTB-MSGDELSTAT-S-002-O	Retrieve Outgoing Message Delivery Status - GET	5.13.3	

B.1.11 SCR for ParlayREST.MMS.Outbound.DeliveryStatus Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-OUTB-DELSTAT-S-001-M	Support for requesting delivery status of outbound messages	5.14	
PARLAYREST-MMS-OUTB-DELSTAT-S-002-M	Retrieve Outgoing Message Delivery Status - GET	5.14.3	

B.1.12 SCR for ParlayREST.MMS.Outbound.Subscriptions Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-OUTB-SUBSCR-S-001-M	Support for outbound subscriptions for a particular client	5.15	
PARLAYREST-MMS-OUTB-SUBSCR-S-002-O	Read all outbound message delivery notification subscriptions - GET	5.15.3	
PARLAYREST-MMS-OUTB-SUBSCR-S-003-M	Create new outbound message subscription – POST (XML and JSON)	5.15.5	
PARLAYREST-MMS-OUTB-SUBSCR-S-003-O	Create new outbound message subscription – POST (www-form-urlencoded)	C.2	

B.1.13 SCR for ParlayREST.MMS.Individual.Outbound.Subscr Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-IND-OUTB-IND-SUBSCR-S-001-M	Support for outbound subscriptions for a particular client	5.16	
PARLAYREST-MMS-IND-OUTB-IND-SUBSCR-S-002-O	Read individual message delivery notification subscription - GET	5.16.3	
PARLAYREST-MMS-IND-OUTB-IND-SUBSCR-S-003-M	Delete subscription for the client - DELETE	5.16.6	

B.1.14 SCR for ParlayREST.MMS.Outbound.DeliveryStatus.Notifications Server

Item	Function	Reference	Requirement
PARLAYREST-MMS-OUTB-DELSTAT-NOTIF-S-001-M	Support for notifying application about delivery status of outbound messages	5.17	
PARLAYREST-MMS-OUTB-DELSTAT-NOTIF-S-002-M	Notify application about delivery status of outbound message - POST	5.17.5	

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

For selected operations, this section defines a format for MMS REST API requests where the body of the **request** is encoded using the *application/x-www-form-urlencoded* MIME type. XML wrapper elements (the root of a complexType) have been omitted from the *x-www-form-urlencoded* model, since it has no hierarchy. Instead the sub-elements of the complex Type are represented in the parameter tables below.

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].

x-www-form-urlencoded bindings for the following MMS REST operations are defined in this section:

- 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.

The request parameters are as follows:

Name	Type/Values	Optional	Description
address	xsd:anyURI [1...unbounded]	No	Destination address(es) for the message
senderAddress	xsd:anyURI	No	The address of the sender to whom a responding message may be sent. If the address is in the form of an MSISDN, include the protocol prefix 'tel' and '%3A%2B' followed by the country code before the subscriber number; e.g. tel%3A%2B19585550100. 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
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 SHOULD use to tag this particular resource representation during a request to create a resource on the server. 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.
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/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
```

```
Content-Length: nnnn
```

```
Accept: application/xml
```

```
Content-Type: multipart/form-data; boundary="=====123456==";
```

```
MIME-Version: 1.0
```

```
Host: example.com:80
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

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

```
Content-Disposition: form-data; name="root-fields"
```

```
Content-Type: application/x-www-form-urlencoded;
```

```
address=tel%3A%2B19585550100&
```

```
address=tel%3A%2B19585550101&
```

```

senderAddress= tel%3A%2B19585550102&
subject= hello%20from%20the%20rest%20of%20us!&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/54311&
clientCorrelator=123456&senderName=Bob
-----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
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550102/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:+19585550100</address>
  <address>tel:+19585550101</address>
  <senderAddress>tel:+19585550102</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/54311</notifyURL>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
  </outboundMMSMessage>
  <clientCorrelator>123456</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550102/requests/req123</resourceURL>
</mms:outboundMessageRequest>

```

C.2 Start delivery receipt notification

This REST method is used by the application to subscribe for the delivery receipt notifications. It **MUST** use the HTTP POST method.

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
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 MAY use to tag this particular resource representation during a request to create a resource on the server. 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.

C.2.1 Example: Create outbound delivery notification subscription (Informative)

C.2.1.1 Request

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept: application/xml
Content-Length: nnnn
Host: example.com:80

filterCriteria=0102&
correlator=123456&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/12345
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<mms:deliveryReceiptSubscription xmlns:mms="urn:oma:xml:rest:messaging:1">
  <callbackReference>
```

```

<notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/12345</notifyURL>
</callbackReference>
<filterCriteria>0102</filterCriteria>
<resourceURL>http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123
</resourceURL>
</mms:deliveryReceiptSubscription>

```

C.3 Start message notification

This REST method is used by the application to subscribe for the notifications of received messages. It MUST use the HTTP POST method.

Request parameters are as follows:

Name	Type/Values	Optional	Description
destinationAddress	xsd:anyURI[1..unbounded]	No	Destination address of the message
criteria	xsd:string	Yes	The text to match against to determine the application to receive the notification
notifyURL	xsd:anyURI	No	Notification endpoint definition
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 MAY use to tag this particular resource representation during a request to create a resource on the server. 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.
useAttachmentURLs	xsd:boolean	Yes	Default: false If set to 'true', inbound message would have links to attachments. 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/1/messaging/inbound/subscriptions HTTP/1.1
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Accept: application/xml
```

```
Content-Length : nnnn
```

```
Host: example.com:80
```

```
destinationAddress=81771&
```

```
criteria=Urgent*&
```

```
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/3455&
```

```
notificationFormat=XML
```

C.3.1.2 Response

```
HTTP/1.1 201 Created
```

```
Content-Type: application/xml
```

```
Content-Length: nnnn
```

```
Location: http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

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

```
<mms:subscription xmlns:mms="urn:oma:xml:rest:messaging:1">
```

```
<callbackReference>
```

```
<notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/3455</notifyURL>
```

```
<notificationFormat>XML</notificationFormat>
```

```
</callbackReference>
```

```
<destinationAddress>81771</destinationAddress>
```

```
<criteria>Urgent*</criteria>
```

```
<resourceURL>http://example.com/exampleAPI/1/messaging/inbound/subscription/sub123</resourceURL>
```

```
<useAttachmentURLs>>false</useAttachmentURLs>
```

```
</mms: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 Parlay REST invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC4627].

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 [OMA_REST_TS_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [OMA_REST_TS_Common].

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

D.1 Retrieve messages for a registration (section 5.4.3.1)

Request:

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"inboundMessageList": {
  "inboundMessage": [
    {
      "destinationAddress": "tel:+19585550100",
      "messageId": "msg123",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123",
      "senderAddress": "tel:+19585550101"
    },
    {
      "destinationAddress": "tel:+19585550102",
      "messageId": "msg124",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg124",
      "senderAddress": "tel:+19585550103"
    }
  ],
  "numberOfMessagesInThisBatch": "2",
  "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2",
  "totalNumberOfPendingMessages": "20"
}}
```

D.2 Request with invalid (non-existing) id (section 5.4.3.2)

Request:

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"requestError": {
  "link": {
    "href": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2",
    "rel": "self"
  },
  "serviceException": {
    "messageId": "SVC0002",
    "text": "Invalid input value. The requested registration id: %1 does not exist",
    "variables": "reg123"
  }
}}
```

D.3 Retrieve messages with attachment URLs (section 5.4.3.3)

Request:

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages?maxBatchSize=2&useAttachmentURLs=true HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"inboundMessageList": {
  "inboundMessage": {
    "destinationAddress": "tel:+19585550100",
    "messageId": "msg123",
    "inboundMMSMessage": {
      "bodyText": "See attached picture",
      "link": [
        {
          "href": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123/attachments/attach123",
          "rel": "attachment"
        }
      ],
    }
  }
}}
```

```

    {
      "href": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123/attachments/attach124",
      "rel": "attachment"
    }
  ],
  "subject": "Who is RESTing on the beach?"
},
"resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123",
"senderAddress": "tel:+19585550101"
},
"numberOfMessagesInThisBatch": "2",
"resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123/attachments",
"totalNumberOfPendingMessages": "20"
}}

```

D.4 Retrieve and delete inbound messages (section 5.5.5.1)

Request:

```

POST /exampleAPI/1/messaging/inbound/registrations/reg123/messages/retrieveAndDeleteMessages HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80

```

```

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

```

Response:

```

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

```

```

{"inboundMessageList": {
  "inboundMessage": [
    {
      "destinationAddress": "tel:+19585550100",
      "messageId": "msg123",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "senderAddress": "tel:+19585550101"
    },
    {
      "destinationAddress": "tel:+19585550102",
      "messageId": "msg124",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "senderAddress": "tel:+19585550103"
    }
  ],
  "numberOfMessagesInThisBatch": "2",

```

```

"resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/retrieveAndDeleteMessages",
"totalNumberOfPendingMessages": "20"
}}

```

D.5 Read and delete one message (section 5.6.5.1)

Request:

```

POST /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/retrieveAndDelete HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80

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

```

Response:

```

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

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

{"inboundMessage": {
  "destinationAddress": "tel:+19585550100",
  "messageId": "msg123",
  "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
  "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123",
  "senderAddress": "tel:+19585550101"
}}

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

Look at the attached picture

-----aaabbb
Content-Disposition: attachments; filename="image1.gif";
Content-Type: image/gif

```

```
MIME-Version: 1.0
Content-ID: <99334422@example.com>
```

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

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

D.6 Read message from gateway storage (section 5.7.3.1)

Request:

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123?resFormat=JSON HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

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

{"inboundMessage": {
  "destinationAddress": "tel:+19585550100",
  "messageId": "msg123",
  "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
  "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123",
  "senderAddress": "tel:+19585550101"
}}

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

Look at the attached picture

-----aaabbb
Content-Disposition: attachments; filename="image1.gif";
Content-Type: image/gif
MIME-Version: 1.0
```

```
Content-ID: <99334422@example.com>
```

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

```
-----aaabbb--
```

```
=====123456=====
```

D.7 Remove message from gateway storage (section 5.7.6.1)

Request:

```
DELETE /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123 HTTP/1.1
```

```
Accept: application/json
```

```
Host: example.com:80
```

Response:

```
HTTP/1.1 204 No content
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.8 Read an MMS attachment (section 5.8.3.1)

Request:

```
GET /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/attachments/{attachmentId} HTTP/1.1
```

```
Accept: image/gif, image/png, image/jpeg, text/html, application/json
```

```
Host: example.com:80
```

Response:

```
HTTP/1.1 200 OK
```

```
Content-Type: image/gif
```

```
Content-Length: nnnn
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

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

D.9 Delete an MMS attachment from gateway storage (section 5.8.6.1)

Request:

```
DELETE /exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123/attachments/{attachmentId} HTTP/1.1
```

```
Accept: application/json
```

```
Host: example.com:80
```

Response:

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

D.10 Read active subscriptions (section 5.9.3.1)

Request:

```
GET /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"subscriptionList": {
  "resourceURL": "http://example.com/exampleAPI/1/messaging/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/1/messaging/inbound/subscriptions/0000001",
      "useAttachmentURLs": "false"
    },
    {
      "callbackReference": {
        "callbackData": "54321",
        "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/54321"
      },
      "clientCorrelator": "567892",
      "criteria": "Urgent*",
      "destinationAddress": "tel:+19585550102",
      "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/0000002",
      "useAttachmentURLs": "false"
    }
  ]
}}
```

D.11 Create inbound subscription (returning a representation of created resource) (section 5.9.5.1)

Request:


```
POST /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80

{"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
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"subscription": {
  "callbackReference": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
  },
  "clientCorrelator": "567893",
  "criteria": "Urgent*",
  "destinationAddress": "tel:+19585550100",
  "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123",
  "useAttachmentURLs": "false"
}}
```

D.12 Create inbound subscription (returning location of created resource) (section 5.9.5.2)

Request:

```
POST /exampleAPI/1/messaging/inbound/subscriptions HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80

{"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
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
{"resourceReference": {"resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123"}}
```

D.13 Read individual subscription (section 5.10.3.1)

Request:

```
GET /exampleAPI/1/messaging/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

```
{"subscription": {
  "callbackReference": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
  },
  "clientCorrelator": "567893",
  "criteria": "Urgent*",
  "destinationAddress": "tel:+19585550100",
  "resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123",
  "useAttachmentURLs": "false"
}}
```

D.14 Delete a subscription (section 5.10.6.1)

Request:

```
DELETE /exampleAPI/1/messaging/inbound/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

D.15 Message arrival notification (section 5.11.5.1)

Request:

```
POST /application.example.com/notifications/DeliveryInfoNotification/88888 HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageNotification": {"inboundMessage": {
  "destinationAddress": "tel:+19585550100",
  "messageId": "msg123",
  "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
  "link": {
    "href": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/sub123",
    "rel": "Subscription"
  },
},
"resourceURL": "http://example.com/exampleAPI/1/messaging/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.16 Message arrival notification with attachment URLs (section 5.11.5.2)

Request:

```
POST /application.example.com/notifications/DeliveryInfoNotification/88888 HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT
Host: application.example.com:80

{"inboundMessageNotification": {"inboundMessage": {
  "destinationAddress": "tel:+19585550100",
  "messageId": "msg123",
  "inboundMMSMessage": {
    "bodyText": "Look at the attached picture",
```

```

"link": [
  {
    "href": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123/attachments/{attachment id1}",
    "rel": "attachment"
  },
  {
    "href": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/msg123/attachments/{attachment id2}",
    "rel": "attachment"
  }
],
"subject": "Who is RESTing on the beach?"
},
"link": {
  "href": "http://example.com/exampleAPI/1/messaging/inbound/subscriptions/{subscriptionId1}",
  "rel": "Subscription"
},
"resourceURL": "http://example.com/exampleAPI/1/messaging/inbound/registrations/reg123/messages/msg123",
"senderAddress": "tel:+19585550101"
}}}

```

Response:

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

D.17 Retrieve list of outgoing requests (section 5.12.3.1)

Request:

```

GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Accept: application/json
Host: example.com:80

```

Response:

```

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

{"outboundMessageRequestList": {"outboundMessageRequest": {
  "address": "tel:+19585550103",
  "clientCorrelator": "567894",
  "deliveryInfoList": {
    "deliveryInfo": {
      "address": "tel:+19585550103",
      "deliveryStatus": "DeliveredToTerminal"
    },
    "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos"
  },
  "outboundMMSMessage": {"subject": "Holiday greetings"},
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
  "senderAddress": "tel:+19585550100"
}}

```

```

},
"resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests"
}}

```

D.18 Create outgoing message, returning the representation of created resource (section 5.12.5.1)

Request:

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Content-Length: nnnn
Accept: application/json
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}

-----123456==

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

====12345====
Content-Disposition: attachments; 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
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"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/99999"
},
"resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
"senderAddress": "tel:+19585550100",
"senderName": "MyName"
}}
```

D.19 Create outgoing message, returning the location of created resource (section 5.12.5.2)

Request:

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="-----123456====";
Accept: application/json
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}

```

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

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

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

```
Content-Disposition: attachments; 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
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

```

{"resourceReference": {"resourceURL": "
http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123"}}

```

D.20 Create outgoing message with charging (section 5.12.5.3)

Request:

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="=====123456====";
Accept: application/json
Content-Length: nnnn

```

```

MIME-Version: 1.0
Host: example.com:80

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

{"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://example-application.com/notifications/DeliveryInfoNotification/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}

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

====12345====
Content-Disposition: attachments; 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
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"requestError": {"policyException": {
  "messageld": "POL0008",

```



```
"text": "Charging is not supported"
}}}
```

D.21 Create outgoing message, serviceException in case of address(es) failure (section 5.12.5.4)

Request:

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/json
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}

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

====12345====
Content-Disposition: attachments; 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": "SVC0001",
  "text": "A service error occurred. Error code is %1 ",
  "variables": "ERROR-XYZ"
}}}
```

D.22 Create outgoing message, multiple addresses partial success, with deliveryInfoList in response (section 5.12.5.5)

Request:

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="=====123456====";
Accept: application/json
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}
```

```
=====123456====
```

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

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

```
Content-Disposition: attachments; 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
```

```
Content-Type: application/json
```

```
Content-Length: nnnn
```

```
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
{
  "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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos\n"
  },
  "outboundMMSMessage": {
    "priority": "High",
    "subject": "hello from the rest of us!"
  },
  "receiptRequest": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/99999"
  },
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}
```

```
}}
```

D.23 Create outgoing message, multiple addresses partial success, without deliveryInfoList in response (section 5.12.5.6)

Request:

```
POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="====123456==";
Accept: application/json
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}

-----123456==

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

====12345===
Content-Disposition: attachments; 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
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"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/99999"
  },
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
  "senderAddress": "tel:+19585550100",
  "senderName": "MyName"
}}
```

D.24 Create outgoing message using SHORT CODE as senderAddress, returning the representation of created resource (section 5.12.5.7)

Request:

```
POST /exampleAPI/1/messaging/outbound/72654/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="-----123456==";
Accept application/json
Content-Length: nnnn
Host: example.com:80
MIME-Version: 1.0

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

{"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/99999"
  },
  "senderAddress": "72654",
  "senderName": "MyName"
}}

```

-----123456==

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

====12345====

Content-Disposition: attachments; 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
 Content-Type: application/json
 Content-Length: nnnn
 Location: http://example.com/exampleAPI/1/messaging/outbound/72654/requests/req123
 Date: Thu, 04 Jun 2009 02:51:59 GMT

```

{"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/99999"
  },
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/72654/requests/req123",
  "senderAddress": "72654",
  "senderName": "MyName"
}}

```

D.25 Read message request and delivery status (section 5.13.3.1)

Request:

```

GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123 HTTP/1.1
Accept: application/json
Host: example.com:80

```

Response:

```

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

{"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/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfos"
},
"outboundMMSMessage": {"subject": "Holiday greetings"},
"receiptRequest": {
  "callbackData": "12345",
  "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/99999"
},
"resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
"senderAddress": "tel:+19585550100",
"senderName": "MyName"
}}

```

D.26 Read message delivery status (section 5.14.3.1)

Request:

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/deliveryInfos HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"deliveryInfoList": {
  "deliveryInfo": [
    {
      "address": "tel:+19585550103",
      "deliveryStatus": "MessageWaiting"
    },
    {
      "address": "tel:+19585550104",
      "deliveryStatus": "MessageWaiting"
    }
  ],
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfo"
}}
```

D.27 Read delivery notification subscriptions (section 5.15.3.1)

Request:

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"deliveryReceiptSubscriptionList": {
  "deliveryReceiptSubscription": [
    {
      "callbackReference": {
        "callbackData": "12345",
        "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/99999"
      },
      "filterCriteria": "0102",
      "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123"
    }
  ]
}}
```



```

    },
    {
      "callbackReference": {
        "callbackData": "54321",
        "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
      },
      "filterCriteria": "0103",
      "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123"
    }
  ],
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/subscriptions"
}}

```

D.28 Create outbound delivery notification subscription (section 5.15.5.1)

Request:

```

POST /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80

```

```

{"deliveryReceiptSubscription": {
  "callbackReference": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/campaign/notifications/DeliveryInfoNotification/77777"
  },
  "filterCriteria": "0102"
}}

```

Response:

```

HTTP/1.1 201 Created
Content-Type: application/json
Content-Length: nnnn
Location: http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

```

{"deliveryReceiptSubscription": {
  "callbackReference": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
  },
  "filterCriteria": "0102",
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123"
}}

```

D.29 Read individual message delivery notification subscription (section 5.16.3.1)

Request:

```
GET /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions/sub123 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

{"deliveryReceiptSubscription": {
  "callbackReference": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/77777"
  },
  "filterCriteria": "0102",
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/subscriptions/sub123"
}}
```

D.30 Delete message delivery notification subscription (section 5.16.6.1)

Request:

```
DELETE /exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/subscriptions /sub123 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

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

D.31 Notify client about outbound message delivery status, multiple delivery status per notification (section 5.17.5.1)

Request:

```
POST http://application/example.com/notifications/DeliveryInfoNotification/77777 HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com:80
```

```
{
  "deliveryInfoList": {
    "deliveryInfo": [
      {
        "address": "tel:+19585550103",
        "deliveryStatus": "DeliveredToTerminal"
      },
      {
        "address": "tel:+19585550104",
        "deliveryStatus": "DeliveredToTerminal"
      }
    ],
    "link": {
      "href": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/{requestId}",
      "rel": "OutboundMessageRequest"
    },
    "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfos"
  }
}
```

Response:

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

D.32 Notify client about outbound message delivery status, single delivery status per notification (section 5.17.5.2)

Request:

```
POST /application/example.com/notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: nnnn
Host: example.com:80

{"deliveryInfoList": {
  "deliveryInfo": {
    "address": "tel:+19585550104",
    "deliveryStatus": "DeliveredToTerminal"
  },
  "link": {
    "href": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123",
    "rel": "self"
  },
  "resourceURL": "http://example.com/exampleAPI/1/messaging/outbound/tel%3A%2B19585550100/requests/req123/DeliveryInfos"
}}
```

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 equivalent operations.

ParlayREST Resource	ParlayREST Method	ParlayREST Section reference	Parlay X equivalent operation
Inbound messages for a given registration	GET	5.4.3	getReceivedMessages 1)
Inbound messages retrieve and delete using registration	POST	5.5.5	getReceivedMessages
Retrieval and deletion of individual inbound message using registration	POST	5.6.5	getMessage
Inbound message for a given registration	GET	5.7.3	getMessage, getMessageURIs 2)
	DELETE	5.7.6	getReceivedMessages
Inbound message subscriptions	POST	5.9.5	startMessageNotification
Individual inbound message subscription	DELETE	5.10.6	stopMessageNotification
Client notification about inbound message	POST	5.11.5	notifyMessageReception
Outbound message requests	POST	5.12.5	sendMessage
Outbound message delivery status	GET	5.14.3	getMessageDeliveryStatus
Outbound message delivery notification subscriptions	POST	5.15.5	startDeliveryReceiptNotification
Individual outbound message delivery notification subscription	DELETE	5.16.6	stopDeliveryReceiptNotification
Client notification about outbound message delivery status	POST	5.17.5	notifyMessageDeliveryReceipt

Table 1: Parlay X operations mapping

- 1) Note: The ParlayX SOAP operation getReceivedMessages is similar to but not quite the same as this ParlayREST 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.