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

5. MESSAGING API DEFINITION .........................................................................................................................................12

   5.1 RESOURCES SUMMARY .........................................................................................................................................13

   5.2 MESSAGING (MMS, SMS, WAP, IM) PARLAY REST 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: InboundSMSMessage ..................................................................................................................................20

       5.2.5 Type: InboundMMSStructure ................................................................................................................................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 ................................................................................................................22

       5.2.11 Type: OutboundMessageRequest .......................................................................................................................23

       5.2.12 Type: OutboundMMSStructure ..........................................................................................................................24

       5.2.13 Type: OutboundWAPMessage ............................................................................................................................25

       5.2.14 Type: OutboundSMSMessage ................................................................................................................................26

       5.2.15 Type: OutboundSMSBinaryMessage ..................................................................................................................26

       5.2.16 Type: OutboundSMSLogoMessage ....................................................................................................................26

       5.2.17 Type: OutboundSMSRingToneMessage ................................................................................................................26

       5.2.18 Type: OutboundIMMessage ................................................................................................................................26

       5.2.19 Type: DeliveryInfoList .......................................................................................................................................27

       5.2.20 Type: DeliveryInfoNotification ........................................................................................................................27

       5.2.21 Type: DeliveryInfo .............................................................................................................................................27

       5.2.22 Type: DeliveryReceiptSubscriptionList .............................................................................................................28

       5.2.23 Type: DeliveryReceiptSubscription ...................................................................................................................29

       5.2.24 Enumeration: DeliveryStatus ................................................................................................................................29

       5.2.25 Enumeration: IMFormat .......................................................................................................................................30

       5.2.26 Enumeration: MessagePriority ...........................................................................................................................30

       5.2.27 Enumeration: RetrievalOrder ................................................................................................................................30

       5.2.28 Enumeration: ServiceIndicationAction ...............................................................................................................30

       5.2.29 Enumeration: ServiceLoadingAction ....................................................................................................................31

       5.2.30 Enumeration: SmsFormat ....................................................................................................................................31

       5.2.31 Enumeration: WAPContent ................................................................................................................................31

       5.2.32 Values of the Link “rel” attribute .......................................................................................................................31

5.3 SEQUENCE DIAGRAMS ..................................................................................................................................................32

       5.3.1 Send message and check the delivery status ........................................................................................................32

       5.3.2 Inbound message delivery (push mode) ................................................................................................................33

       5.3.3 Inbound message delivery (polling mode) ................................................................................................................34

5.4 RESOURCE: INBOUND MESSAGES FOR A GIVEN REGISTRATION ..............................................................................36

       5.4.1 Request URI variables ...........................................................................................................................................36

       5.4.2 Response Codes ....................................................................................................................................................36
5.4.3 GET .................................................................................................................................................................... 36
5.4.4 PUT .................................................................................................................................................................... 39
5.4.5 POST .................................................................................................................................................................. 39
5.4.6 DELETE ............................................................................................................................................................ 39
5.5 RESOURCE: INBOUND MESSAGES RETRIEVE AND DELETE USING REGISTRATION .......................................................... 39
  5.5.1 Request URI variables ....................................................................................................................................... 39
  5.5.2 Response Codes ............................................................................................................................................... 40
  5.5.3 GET ............................................................................................................................................................... 40
  5.5.4 PUT ............................................................................................................................................................... 40
  5.5.5 POST ............................................................................................................................................................. 40
  5.5.6 DELETE .......................................................................................................................................................... 40
5.6 RESOURCE: RETRIEVAL AND DELETION OF INDIVIDUAL INBOUND MESSAGE USING REGISTRATION ......................... 41
  5.6.1 Request URI variables ..................................................................................................................................... 42
  5.6.2 Response Codes .............................................................................................................................................. 42
  5.6.3 GET ............................................................................................................................................................... 42
  5.6.4 POST ............................................................................................................................................................. 42
  5.6.5 PUT ............................................................................................................................................................... 43
  5.6.6 DELETE .......................................................................................................................................................... 44
5.7 RESOURCE: INBOUND MESSAGE FOR A GIVEN REGISTRATION .................................................................................. 44
  5.7.1 Request URI variables ..................................................................................................................................... 44
  5.7.2 Response Codes .............................................................................................................................................. 44
  5.7.3 GET ............................................................................................................................................................... 44
  5.7.4 PUT ............................................................................................................................................................... 45
  5.7.5 POST ............................................................................................................................................................. 46
  5.7.6 DELETE .......................................................................................................................................................... 46
5.8 RESOURCE: INBOUND MESSAGE ATTACHMENT ........................................................................................................ 46
  5.8.1 Request URI variables ..................................................................................................................................... 46
  5.8.2 Response Codes .............................................................................................................................................. 47
  5.8.3 GET ............................................................................................................................................................... 47
  5.8.4 PUT ............................................................................................................................................................... 47
  5.8.5 POST ............................................................................................................................................................. 47
  5.8.6 DELETE .......................................................................................................................................................... 47
5.9 RESOURCE: INBOUND MESSAGE SUBSCRIPTIONS .................................................................................................... 48
  5.9.1 Request URI variables ..................................................................................................................................... 48
  5.9.2 Response Codes .............................................................................................................................................. 48
  5.9.3 GET ............................................................................................................................................................... 48
  5.9.4 PUT ............................................................................................................................................................... 49
  5.9.5 POST ............................................................................................................................................................. 49
  5.9.6 DELETE .......................................................................................................................................................... 49
5.10 RESOURCE: INDIVIDUAL INBOUND MESSAGE SUBSCRIPTION ..................................................................................... 51
  5.10.1 Request URI variables ..................................................................................................................................... 51
  5.10.2 Response Codes .............................................................................................................................................. 51
  5.10.3 GET ............................................................................................................................................................... 51
  5.10.4 PUT ............................................................................................................................................................... 52
  5.10.5 POST ............................................................................................................................................................. 52
  5.10.6 DELETE .......................................................................................................................................................... 52
5.11 RESOURCE: CLIENT NOTIFICATION ABOUT INBOUND MESSAGE ..................................................................................... 53
  5.11.1 Request URI variables ..................................................................................................................................... 53
  5.11.2 Response Codes .............................................................................................................................................. 53
  5.11.3 GET ............................................................................................................................................................... 53
  5.11.4 PUT ............................................................................................................................................................... 53
  5.11.5 POST ............................................................................................................................................................. 53
  5.11.6 DELETE .......................................................................................................................................................... 55
5.12 RESOURCE: OUTBOUND MESSAGE REQUESTS ........................................................................................................ 55
  5.12.1 Request URI variables ..................................................................................................................................... 55
  5.12.2 Response Codes .............................................................................................................................................. 55
  5.12.3 GET ............................................................................................................................................................... 55
**APPENDIX A. CHANGE HISTORY (INFORMATIVE)**

A.1 APPROVED VERSION HISTORY

A.2 DRAFT/CANDIDATE VERSION 1.0 HISTORY

**APPENDIX B. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE)**

B.1 SCR FOR PARLAYREST.MMS SERVER

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

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

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

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

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

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

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

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

B.1.9 SCR for ParlayREST.MMS.Outbound Server

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

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

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

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

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document

APPENDIX C. APPLICATION/X-WWW-FORM-URLENCODED REQUEST FORMAT FOR SELECTED OPERATIONS

C.1 SEND A MESSAGE TO A TERMINAL
C.1.1 Example: Create outgoing message (Informative)
C.2 START DELIVERY RECEIPT NOTIFICATION
C.2.1 Example: Create outbound delivery notification subscription (Informative)
C.3 START MESSAGE NOTIFICATION
C.3.1 Example: Create inbound subscription (Informative)

APPENDIX D. JSON EXAMPLES (INFORMATIVE)

D.1 RETRIEVE MESSAGES FOR A REGISTRATION (SECTION 5.4.3.1)
D.2 REQUEST WITH INVALID (NON-EXISTING) ID (SECTION 5.4.3.2)
D.3 RETRIEVE MESSAGES WITH ATTACHMENT URLS (SECTION 5.4.3.3)
D.4 RETRIEVE AND DELETE INBOUND MESSAGES (SECTION 5.5.5.1)
D.5 READ AND DELETE ONE MESSAGE (SECTION 5.6.4.1)
D.6 READ MESSAGE FROM GATEWAY STORAGE (SECTION 5.7.3.1)
D.7 REMOVE MESSAGE FROM GATEWAY STORAGE (SECTION 5.7.6.1)
D.8 READ AN MMS ATTACHMENT (SECTION 5.8.3.1)
D.9 DELETE AN MMS ATTACHMENT FROM GATEWAY STORAGE (SECTION 5.8.6.1)
D.10 READ ACTIVE SUBSCRIPTIONS (SECTION 5.9.3.1)
D.11 CREATE INBOUND SUBSCRIPTION (RETURNING A REPRESENTATION OF CREATED RESOURCE) (SECTION 5.9.5.1)
D.12 CREATE INBOUND SUBSCRIPTION (RETURNING LOCATION OF CREATED RESOURCE) (SECTION 5.9.5.2)
D.13 READ INDIVIDUAL SUBSCRIPTION (SECTION 5.10.3.1)
D.14 DELETE A SUBSCRIPTION (SECTION 5.10.6.1)
D.15 MESSAGE ARRIVAL NOTIFICATION (SECTION 5.11.5.1)
D.16 MESSAGE ARRIVAL NOTIFICATION WITH ATTACHMENT URLS (SECTION 5.11.5.2)
D.17 RETRIEVE LIST OF OUTGOING REQUESTS (SECTION 5.12.3.1)
D.18 CREATE OUTGOING MESSAGE, RETURNING THE REPRESENTATION OF CREATED RESOURCE (SECTION 5.12.5.1)
D.19 CREATE OUTGOING MESSAGE, RETURNING THE LOCATION OF CREATED RESOURCE (SECTION 5.12.5.2)
D.20 CREATE OUTGOING MESSAGE WITH CHARGING (SECTION 5.12.5.3)
D.21 CREATE OUTGOING MESSAGE, SERVICE EXCEPTION IN CASE OF ADDRESS(ES) FAILURE (SECTION 5.12.5.4)
D.22 CREATE OUTGOING MESSAGE, MULTIPLE ADDRESSES PARTIAL SUCCESS, WITH DELIVERY INFO LIST IN RESPONSE (SECTION 5.12.5.5)
D.23 CREATE OUTGOING MESSAGE, MULTIPLE ADDRESSES PARTIAL SUCCESS, WITHOUT DELIVERY INFO LIST IN RESPONSE (SECTION 5.12.5.6)
D.24 READ MESSAGE REQUEST AND DELIVERY STATUS (SECTION 5.13.3.1)
D.25 READ MESSAGE DELIVERY STATUS (SECTION 5.14.3.1)
D.26 READ DELIVERY NOTIFICATION SUBSCRIPTIONS (SECTION 5.15.3.1)
D.27 CREATE OUTBOUND DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.15.5.1)
D.28 READ INDIVIDUAL MESSAGE DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.16.3.1)
D.29 DELETE MESSAGE DELIVERY NOTIFICATION SUBSCRIPTION (SECTION 5.16.6.1)
D.30 NOTIFY CLIENT ABOUT OUTBOUND MESSAGE DELIVERY STATUS, MULTIPLE DELIVERY STATUS PER NOTIFICATION (SECTION 5.17.5.1)
D.31 NOTIFY CLIENT ABOUT OUTBOUND MESSAGE DELIVERY STATUS, SINGLE DELIVERY STATUS PER NOTIFICATION (SECTION 5.17.5.2)

APPENDIX E. PARLAY X OPERATIONS MAPPING (INFORMATIVE)

Figures

Figure 1 Resource structure defined by this specification
Figure 2 Send message and check the delivery status
Figure 3 Inbound message delivery (push mode) .................................................................................................................. 34
Figure 4 Inbound message delivery (polling mode) .............................................................................................................. 35
1. Scope

This specification defines an HTTP protocol binding for an abstract API using the REST architectural style, based on existing OMA enabler namely the Multi Media Messaging Service, as defined in [3GPP 29.199-5].
2. References

2.1 Normative References


[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


2.2 Informative References


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

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>EMS</td>
<td>Enhanced Message Service</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>IM</td>
<td>Instant Message</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>MIME</td>
<td>Multipurpose Internet Mail Extensions</td>
</tr>
<tr>
<td>MMS</td>
<td>Multi Media System</td>
</tr>
<tr>
<td>MSISDN</td>
<td>Mobile Subscriber ISDN Number</td>
</tr>
<tr>
<td>OMA</td>
<td>Open Mobile Alliance</td>
</tr>
<tr>
<td>REST</td>
<td>REpresentational State Transfer</td>
</tr>
<tr>
<td>RTX</td>
<td>Ring Tone eXtended</td>
</tr>
<tr>
<td>SCR</td>
<td>Static Conformance Requirements</td>
</tr>
<tr>
<td>SI</td>
<td>Service Indication</td>
</tr>
<tr>
<td>SL</td>
<td>Service Logic</td>
</tr>
<tr>
<td>SMS</td>
<td>Short Message Service</td>
</tr>
<tr>
<td>TS</td>
<td>Technical Specification</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>WAP</td>
<td>Wireless Application Protocol</td>
</tr>
<tr>
<td>WP</td>
<td>White Paper</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition</td>
</tr>
</tbody>
</table>
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)
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].

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 [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.

![Resource structure defined by this specification](image)

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)

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>GET</td>
</tr>
<tr>
<td>Inbound messages for a given registration</td>
<td>/inbound/registrations/{registrationId}/messages</td>
<td>InboundMessageList</td>
<td>no</td>
</tr>
<tr>
<td>Note: Used by clients that periodically poll for incoming messages. Retrieval criteria have to be provisioned in advance.</td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>Inbound messages retrieve and delete using registration</td>
<td>/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages</td>
<td>InboundMessageList (used for POST response)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InboundMessageRetrieveAndDeleteRequest (used for POST request)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>Retrieval and deletion of individual inbound message using registration</td>
<td>/inbound/registrations/{registrationId}/messages/{messageId}/retrieveAndDelete</td>
<td>InboundMessage (used for POST response)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InboundMessageRetrieveAndDeleteRequest (used for POST request)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>Inbound message for a given registration</td>
<td>/inbound/registrations/{registrationId}/messages/{messageId}</td>
<td>InboundMessage</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>read one message from gateway storage</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>delete one message from gateway storage</td>
<td>no</td>
</tr>
<tr>
<td>Resource</td>
<td>URL Base URL: http://{serverRoot}/{apiVersion}/messaging</td>
<td>Data Structures</td>
<td>HTTP verbs</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------------</td>
<td>-----------------</td>
<td>------------</td>
</tr>
<tr>
<td>Inbound message attachment</td>
<td>/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId}</td>
<td>Any MIME content (the one of the attachment)</td>
<td>read individual message attachment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Purpose:** Subscription Management for Inbound messages

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/{apiVersion}/messaging</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound message subscriptions</td>
<td>/inbound/subscriptions</td>
<td>SubscriptionList (used for GET) Subscription (used for POST) common:ResourceReference (optional alternative for POST response)</td>
<td>read all active subscriptions</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Individual inbound message subscription</td>
<td>/inbound/subscriptions/{subscriptionId}</td>
<td>Subscription</td>
<td>read individual subscription</td>
</tr>
</tbody>
</table>

**Purpose:** Callback notifications for Inbound Messages
### Purpose: Sending message and obtaining the Delivery Status

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/{apiVersion}/messaging</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
<th>GET</th>
<th>POST</th>
<th>PUT</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Outbound message requests</strong></td>
<td>/outbound/{senderAddress}/requests</td>
<td>OutboundMessageRequestList (used for GET)</td>
<td>read all pending outbound message requests with current delivery status</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OutboundMessageRequest (used for POST)</td>
<td>create new outbound message request</td>
<td>no</td>
<td>no</td>
<td></td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>common:ResourceReference (optional alternative for POST response)</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td><strong>Outbound message request and delivery status</strong></td>
<td>/outbound/{senderAddress}/requests/{requestId}</td>
<td>OutboundMessageRequest</td>
<td>read message and delivery status for the individual outbound message request</td>
<td>no</td>
<td>no</td>
<td></td>
<td>no</td>
</tr>
<tr>
<td><strong>Outbound message delivery status</strong></td>
<td>/outbound/{senderAddress}/requests/{requestId}/deliveryInfos</td>
<td>DeliveryInfoList</td>
<td>read delivery status for the individual outbound message request</td>
<td>no</td>
<td>no</td>
<td></td>
<td>no</td>
</tr>
</tbody>
</table>
Purpose: Subscription Management for Outbound Message Delivery Status (overwrites individual request notifications)

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL Base URL: http://{serverRoot}/{apiVersion}/messaging</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outbound message delivery notification subscriptions</td>
<td>/outbound/{senderAddress}/subscriptions</td>
<td>DeliveryReceiptSubscriptionList (used for GET)</td>
<td>read all outbound message subscriptions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeliveryReceiptSubscription (used for POST)</td>
<td>create new delivery receipt subscription</td>
</tr>
<tr>
<td></td>
<td></td>
<td>common:ResourceReference (optional alternative for POST response)</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>Individual outbound message delivery notification subscription</td>
<td>/outbound/{senderAddress}/subscriptions/{subscriptionId}</td>
<td>DeliveryReceiptSubscription</td>
<td>read an individual outbound message subscription</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>no</td>
</tr>
</tbody>
</table>

Purpose: Callback notifications for Outbound Message Delivery Status

<table>
<thead>
<tr>
<th>Resource</th>
<th>URL &lt;specified by the client&gt;</th>
<th>Data Structures</th>
<th>HTTP verbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client notification about outbound message delivery status</td>
<td>&lt;specified by the client when outbound request is submitted&gt;</td>
<td>DeliveryInfoNotification</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
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

Received message list.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inboundMessage</td>
<td>InboundMessage [0..unbounded]</td>
<td>Yes</td>
<td>It may contain an array of messages received according to the specified registrationid.</td>
</tr>
<tr>
<td>totalNumberOfPendingMessages</td>
<td>xsd:int</td>
<td>Yes</td>
<td>Total number of messages in the gateway storage waiting for retrieval at the time of the request</td>
</tr>
<tr>
<td>numberOfMessagesInThisBatch</td>
<td>xsd:int</td>
<td>Yes</td>
<td>Number of the messages included in the response (part of the totalNumberOfPendingMessages)</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body, and in PUT requests.</td>
</tr>
</tbody>
</table>

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

5.2.2 Type: InboundMessage

Individual incoming message.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destinationAddress</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Number associated with the invoked Message service, i.e. the destination address used by the terminal to send the message.</td>
</tr>
<tr>
<td>senderAddress</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Indicates message senderAddress.</td>
</tr>
<tr>
<td>dateTime</td>
<td>xsd:dateTime</td>
<td>Yes</td>
<td>Time when message was received by operator</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body, and in PUT requests.</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources that are in relationship with the resource</td>
</tr>
<tr>
<td>messageId</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>inboundSMSTextMessage</td>
<td>InboundSMSTextMessage</td>
<td>Choice</td>
<td>Inbound SMS Text Message</td>
</tr>
<tr>
<td>inboundMMSMessage</td>
<td>InboundMMSMessage</td>
<td>Choice</td>
<td>Inbound MMS Message</td>
</tr>
<tr>
<td>inboundIMMessage</td>
<td>InboundIMMessage</td>
<td>Choice</td>
<td>Inbound IM Message</td>
</tr>
</tbody>
</table>

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

### 5.2.3 Type: InboundMessageNotification

Notification carrying an individual incoming message.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackData</td>
<td>xsd:string</td>
<td>Yes</td>
<td>CallbackData as passed by the application during the associated Send MMS operation. See [REST_TS_Common], section 6.2.4.</td>
</tr>
<tr>
<td>inboundMessage</td>
<td>InboundMessage</td>
<td>No</td>
<td>Multimedia message</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources. For example we can have a link to the subscription used to receive this message.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>xsd:string</td>
<td>No</td>
<td>Short message content.</td>
</tr>
</tbody>
</table>

5.2.5 Type: InboundMMSMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>xsd:string</td>
<td>Yes</td>
<td>If present, indicates the subject of the received message.</td>
</tr>
<tr>
<td>priority</td>
<td>MessagePriority</td>
<td>Yes</td>
<td>The priority of the message: default is Normal.</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources (like individual attachments: &lt;Link rel=&quot;attachment&quot; href=&quot;../inbound/registration/{registrationId}/messages/{messageId}/attachments/{attachmentId}&quot;/&gt;)</td>
</tr>
<tr>
<td>bodyText</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Contains the message body if it is encoded as ASCII text</td>
</tr>
</tbody>
</table>

5.2.6 Type: InboundIMMessage

<table>
<thead>
<tr>
<th>Element name</th>
<th>Element type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>xsd:string</td>
<td>Yes</td>
<td>If present, indicates the subject of the received IM message.</td>
</tr>
<tr>
<td>priority</td>
<td>MessagePriority</td>
<td>Yes</td>
<td>The priority of the message: default is Normal.</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources (like individual attachments: &lt;Link rel=&quot;attachment&quot; href=&quot;../inbound/registration/{registrationId}/messages/{messageId}/attachments/{attachmentId}&quot;/&gt;)</td>
</tr>
<tr>
<td>imFormat</td>
<td>IMFormat</td>
<td>Yes</td>
<td>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)</td>
</tr>
<tr>
<td>bodyText</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Contains the message body if it is encoded as ASCII text</td>
</tr>
</tbody>
</table>
### 5.2.7 Type: SubscriptionList

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subscription</td>
<td>Subscription[0..unbounded]</td>
<td>Yes</td>
<td>It may contain an array of Subscription</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body, and in PUT requests.</td>
</tr>
</tbody>
</table>

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

### 5.2.8 Type: Subscription

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackReference</td>
<td>common:CallbackReference</td>
<td>No</td>
<td>Client's Notification endpoint and parameters</td>
</tr>
<tr>
<td>destinationAddress</td>
<td>xsd:anyURI [1…unbounded]</td>
<td>No</td>
<td>the destination address of the multimedia message</td>
</tr>
<tr>
<td>criteria</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The text to match against to determine the application to receive the notification. 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. If the subject of the multimedia 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]</td>
</tr>
<tr>
<td>clientCorrelator</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body.</td>
</tr>
</tbody>
</table>
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 section 5.6.1 of [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. Section 5.6.1 of [REST_TS_Common] provides a recommendation regarding the generation of the value of this field.

### 5.2.9 Type: InboundMessageRetrieveAndDeleteRequest

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>retrievalOrder</td>
<td>RetrievalOrder</td>
<td>Yes</td>
<td>Specifies order in which messages should be retrieved if there are more than one pending</td>
</tr>
<tr>
<td>priority</td>
<td>MessagePriority</td>
<td>Yes</td>
<td>The priority of the message: default is Normal.</td>
</tr>
<tr>
<td>maxBatchSize</td>
<td>xsd:int</td>
<td>Yes</td>
<td>Specifies maximum number of messages to be returned in the response</td>
</tr>
<tr>
<td>useAttachmentURLs</td>
<td>xsd:boolean</td>
<td>No</td>
<td>If set to ‘true’, inbound message will have links to attachments. Otherwise, inbound message includes attachments using MIME</td>
</tr>
</tbody>
</table>

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

### 5.2.10 Type: OutboundMessageRequestList

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outboundMessageRequest</td>
<td>OutboundMessageRequest</td>
<td>Yes</td>
<td>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</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL</td>
</tr>
</tbody>
</table>
A root element named `outboundMessageRequestList` of type `OutboundMessageRequestList` is allowed in request and/or response bodies.

### 5.2.11 Type: OutboundMessageRequest

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>xsd:anyURI [1..unbounded]</td>
<td>No</td>
<td>Destination addresses for the Message.</td>
</tr>
<tr>
<td>senderAddress</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>The address of the sender to whom a responding SMS may be sent. If the address is in the form of an MSISDN, include the protocol prefix 'tel:' and '%2B' followed by the country code before the subscriber number; e.g. tel:%2B447990123456.</td>
</tr>
<tr>
<td>senderName</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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).</td>
</tr>
<tr>
<td>charging</td>
<td>common:Charging Information</td>
<td>Yes</td>
<td>Charging to apply to this message.</td>
</tr>
<tr>
<td>receiptRequest</td>
<td>common:CallbackReference</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>outboundSMSTextMessage</td>
<td>OutboundSMSTextMessage</td>
<td>Choice</td>
<td>Included if a SMSText is being sent.</td>
</tr>
<tr>
<td>outboundSMSBinaryMessage</td>
<td>OutboundSMSBinaryMessage</td>
<td>Choice</td>
<td>Included if a SMS Binary is being sent.</td>
</tr>
<tr>
<td>outboundSMSLogoMessage</td>
<td>OutboundSMSLogoMessage</td>
<td>Choice</td>
<td>Included if a SMSLogo is being sent.</td>
</tr>
<tr>
<td>outboundSMSRingToneMessage</td>
<td>OutboundSMSRingToneMessage</td>
<td>Choice</td>
<td>Included if a SMSRingtone is being sent.</td>
</tr>
<tr>
<td>outboundWAPMessage</td>
<td>OutboundWAPMessage</td>
<td>Choice</td>
<td>Included if WAP is being used.</td>
</tr>
<tr>
<td>outboundMMSMessage</td>
<td>OutboundMMSMessage</td>
<td>Choice</td>
<td>Included if MMS is being sent.</td>
</tr>
<tr>
<td>outboundIMMessage</td>
<td>OutboundIMMessage</td>
<td>Choice</td>
<td>Included if IM is being sent.</td>
</tr>
<tr>
<td>clientCorrelator</td>
<td>xsd:string</td>
<td>Yes</td>
<td>A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server.</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body, and in PUT requests.</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources that are in relationship with the resource</td>
</tr>
<tr>
<td>deliveryInfoList</td>
<td>DeliveryInfoList</td>
<td>Yes</td>
<td>The Delivery Information (filled in by the server)</td>
</tr>
</tbody>
</table>

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

Note that the clientCorrelator is used for purposes of error recovery as specified in section 5.6.1 of [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. Section 5.6.1 of [REST_TS_Common] provides a recommendation regarding the generation of the value of this field.

### 5.2.12 Type: OutboundMMSMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>xsd:string</td>
<td>Yes</td>
<td>If present, indicates the subject of the received message.</td>
</tr>
<tr>
<td>priority</td>
<td>MessagePriority</td>
<td>Yes</td>
<td>The priority of the message: default is Normal.</td>
</tr>
</tbody>
</table>
### 5.2.13 Type: OutboundWAPMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentType</td>
<td>WAPContent</td>
<td>No</td>
<td>The type of content delivery notification to send.</td>
</tr>
<tr>
<td>targetURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>A URL from which content may be loaded by a terminal</td>
</tr>
<tr>
<td>serviceLoadingAction</td>
<td>ServiceLoadingAction</td>
<td>Choice</td>
<td>There is no user intervention.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If the parameter is not specified, the default value will be &quot;ExecuteLow&quot;.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>See [WAP-SL] for more details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>May be present only if ContentType is “ServiceLoading”</td>
</tr>
<tr>
<td>serviceIndicationAction</td>
<td>ServiceIndicationAction</td>
<td>Choice</td>
<td>Allows controlling the level of intrusiveness, of outbound wap push messages.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>According to [WAP-SI] it contains a text string specifying the action to be</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>taken when the message is received.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If the parameter is not specified, the value “SignalMedium” is used</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>May be resent only if ContentType is “ServiceIndication”</td>
</tr>
<tr>
<td>text</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Information that accompanies the push.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>May be present only if ServiceIndicationAction is present and ContentType is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>“ServiceIndication”</td>
</tr>
<tr>
<td>created</td>
<td>xsd:dateTime</td>
<td>Yes</td>
<td>This attribute may be used to specify the date and time associated with the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>creation or last modification of the content indicated by targetURL, which</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>may differ from the date and time when the message was created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>May be present only if ContentType is “ServiceIndication”.</td>
</tr>
</tbody>
</table>

XSD modelling uses a “choice” to select either a serviceLoadingAction or serviceIndicationAction plus text and created.
### 5.2.14 Type: OutboundSMSTextMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>xsd:string</td>
<td>No</td>
<td>Short message content.</td>
</tr>
</tbody>
</table>

### 5.2.15 Type: OutboundSMSBinaryMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>xsd:base64Binary</td>
<td>No</td>
<td>Short message content in binary format.</td>
</tr>
</tbody>
</table>

### 5.2.16 Type: OutboundSMSLogoMessage

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

### 5.2.17 Type: OutboundSMSRingToneMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ringTone</td>
<td>xsd:string</td>
<td>No</td>
<td>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.</td>
</tr>
<tr>
<td>smsFormat</td>
<td>SmsFormat</td>
<td>No</td>
<td>Conversion to be applied to the message prior to delivery. Possible values are: 'Ems' or 'SmartMessaging'</td>
</tr>
</tbody>
</table>

### 5.2.18 Type: OutboundIMMessage

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>xsd:string</td>
<td>Yes</td>
<td>If present, indicates the subject of</td>
</tr>
</tbody>
</table>
the received message.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imFormat</td>
<td>IMFormat</td>
<td>Yes</td>
<td>The type of IM</td>
</tr>
<tr>
<td>bodyText</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Contains the message body if it is encoded as ASCII text</td>
</tr>
</tbody>
</table>

### 5.2.19 Type: DeliveryInfoList

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Self referring URL</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Linked to other resources that are in relationship with the resource</td>
</tr>
<tr>
<td>deliveryInfo</td>
<td>DeliveryInfo[1..unbounded]</td>
<td>No</td>
<td>Delivery Information</td>
</tr>
</tbody>
</table>

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

### 5.2.20 Type: DeliveryInfoNotification

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackData</td>
<td>xsd:string</td>
<td>Yes</td>
<td>CallbackData if passed by the application in the receiptRequest element during the associated Send message operation. See [REST_TS_Common], section 6.2.4.</td>
</tr>
<tr>
<td>deliveryInfo</td>
<td>DeliveryInfo[1..unbounded]</td>
<td>No</td>
<td>Delivery Information</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources. For example we can have a link to the original outbound message request.</td>
</tr>
</tbody>
</table>

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

### 5.2.21 Type: DeliveryInfo

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Outbound message destination address</td>
</tr>
<tr>
<td>deliveryStatus</td>
<td>DeliveryStatus</td>
<td>No</td>
<td>Indicates the delivery result for the destination address.</td>
</tr>
<tr>
<td>description</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Used together with Delivery Status (e.g. DeliveryImpossible) to provide</td>
</tr>
</tbody>
</table>
5.2.22 Type: DeliveryReceiptSubscriptionList

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Self referring URL</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources that are in relationship with the resource</td>
</tr>
<tr>
<td>deliveryReceiptSubscription</td>
<td>DeliveryReceiptSubscription[0...unbounded]</td>
<td>Yes</td>
<td>Delivery Information</td>
</tr>
</tbody>
</table>

A root element named deliveryReceiptSubscriptionList of type DeliveryReceiptSubscriptionList is allowed in request and/or response bodies.
### 5.2.23 Type: DeliveryReceiptSubscription

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackReference</td>
<td>common:CallbackReference</td>
<td>No</td>
<td>Notification endpoint and parameters definition</td>
</tr>
<tr>
<td>filterCriteria</td>
<td>xsd:string</td>
<td>No</td>
<td>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.</td>
</tr>
<tr>
<td>clientCorrelator</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>resourceURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>Self referring URL. SHALL NOT be included in POST requests, MUST be included in responses to any HTTP method that returns an entity body, and in PUT requests.</td>
</tr>
<tr>
<td>link</td>
<td>common:Link[0..unbounded]</td>
<td>Yes</td>
<td>Link to other resources that are in relationship with the resource</td>
</tr>
</tbody>
</table>

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

Note that the clientCorrelator is used for purposes of error recovery as specified in section 5.6.1 of [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. Section 5.6.1 of [REST_TS_Common] provides a recommendation regarding the generation of the value of this field.

### 5.2.24 Enumeration: DeliveryStatus

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeliveredToTerminal</td>
<td>Successful delivery to Terminal.</td>
</tr>
<tr>
<td>DeliveryUncertain</td>
<td>Delivery status unknown: e.g. because it was handed off to another network.</td>
</tr>
<tr>
<td>DeliveryImpossible</td>
<td>Unsuccessful delivery; the message could not be delivered before it expired.</td>
</tr>
<tr>
<td>MessageWaiting</td>
<td>The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.</td>
</tr>
<tr>
<td>DeliveredToNetwork</td>
<td>Successful delivery to the network enabler responsible for distributing the multimedia message further in the network.</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IM</td>
<td>Instant (immediate) messaging service (Can be short IM or large IM. Underlying network can decide message type from message context)</td>
</tr>
<tr>
<td>IMPagerMode</td>
<td>Short IM text message, as defined in [OMA-IM-TS].</td>
</tr>
<tr>
<td>IMLargeMode</td>
<td>Large IM message with multimedia, as defined in [OMA-IM-TS].</td>
</tr>
<tr>
<td>IMFFileTransfer</td>
<td>Large IM used for File Transfer, as defined in [OMA-IM-TS]</td>
</tr>
</tbody>
</table>

### 5.2.26 Enumeration: MessagePriority

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Default message priority</td>
</tr>
<tr>
<td>Low</td>
<td>Low message priority</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal message priority</td>
</tr>
<tr>
<td>High</td>
<td>High message priority</td>
</tr>
</tbody>
</table>

### 5.2.27 Enumeration: RetrievalOrder

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OldestFirst</td>
<td>Retrieve in the order from oldest to newest</td>
</tr>
<tr>
<td>NewestFirst</td>
<td>Retrieve in the order from newest to oldest</td>
</tr>
</tbody>
</table>

### 5.2.28 Enumeration: ServiceIndicationAction

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SignalNone</td>
<td>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].</td>
</tr>
<tr>
<td>SignalLow</td>
<td>The SI MUST be postponed without user intervention</td>
</tr>
<tr>
<td>SignalMedium</td>
<td>The SI MUST be presented as soon as the implementation allows that to be carried out in a non-user-intrusive manner.</td>
</tr>
<tr>
<td>SignalHigh</td>
<td>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.</td>
</tr>
<tr>
<td>Delete</td>
<td>The message should be discarded.</td>
</tr>
</tbody>
</table>
### 5.2.29 Enumeration: ServiceLoadingAction

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExecuteLow</td>
<td>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]</td>
</tr>
<tr>
<td>ExecuteHigh</td>
<td>The service is loaded and executed in the same way as for ExecuteLow, but MAY result in a user-intrusive behavior.</td>
</tr>
<tr>
<td>Cache</td>
<td>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.</td>
</tr>
</tbody>
</table>

### 5.2.30 Enumeration: SmsFormat

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ems</td>
<td>EMS conversion</td>
</tr>
<tr>
<td>SmartMessaging</td>
<td>SmartMessaging conversion</td>
</tr>
</tbody>
</table>

### 5.2.31 Enumeration: WAPContent

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceIndication</td>
<td>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]</td>
</tr>
<tr>
<td>ServiceLoading</td>
<td>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]</td>
</tr>
</tbody>
</table>

### 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
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.

### 5.3 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
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
  \[http://\{serverRoot\} /{apiVersion}/messaging/inbound/subscriptions/{subscriptionId}\]
Outline of the flows:

1. An application subscribes to notifications for inbound messages using POST and receives the resulting resource URL containing the subscription ID.

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 reads the attached content using attachment URL in the message request.

4. The application stops the notifications subscription using DELETE with a resource URL containing the subscription ID.

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 a 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.
2. An application requests the list of the incoming messages fulfilling specified criteria using GET with a resource URL containing the registrationId.
3. The application reads one message request using GET with a resource URL containing the messageId.
4. The application reads one attachment to the message using GET with a resource URL containing the attachmentId.
5. 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients wants to use</td>
</tr>
<tr>
<td>registrationId</td>
<td>reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier</td>
</tr>
</tbody>
</table>

5.4.2 Response Codes

5.4.2.1 HTTP Response Codes

For HTTP response codes, see [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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxBatchSize</td>
<td>xsd:int</td>
<td>Yes</td>
<td>Specifies maximum number of messages to be returned in the response</td>
</tr>
<tr>
<td>retrievalOrder</td>
<td>RetrievalOrder</td>
<td>Yes</td>
<td>Specifies order in which messages should be retrieved is there are more then one pending</td>
</tr>
<tr>
<td>useAttachmentURLs</td>
<td>xsd:boolean</td>
<td>Yes</td>
<td>Default: false</td>
</tr>
</tbody>
</table>
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 .../{apiVersion}/messaging/inbound/registrations/{registrationId}/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: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

```xml
<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageList xmlns:mms="urn:oma:xml:rest:messaging:1">
  <!-- MMS -->
  <inboundMessage>
    <destinationAddress>MSISDN1</destinationAddress>
    <senderAddress>MSISDN2</senderAddress>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}</resourceURL>
    <messageId>{messageId1}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
  <!-- MMS -->
  <inboundMessage>
    <destinationAddress>MSISDN3</destinationAddress>
    <senderAddress>MSISDN4</senderAddress>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId2}</resourceURL>
    <messageId>{messageId2}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
  <totalNumberOfPendingMessages>20</totalNumberOfPendingMessages>
  <numberOfMessagesInThisBatch>2</numberOfMessagesInThisBatch>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages?maxBatchSize=2</resourceURL>
</mms:inboundMessageList>
```
5.4.3.2 Example 2: request with invalid (non-existing) id

5.4.3.2.1 Request

GET .../(apiVersion)/messaging/inbound/registrations/registration123/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: 12345
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://{serverRoot}/{apiVersion}/messaging/inbound/registrations/registration123/messages?maxBatchSize=2" />
  <serviceException>
    <messageId>SVC0002</messageId>
    <text>Invalid input value. The requested registration id: %1 does not exist</text>
    <variables>registration123</variables>
  </serviceException>
</common:requestError>

5.4.3.3 Example 3: Retrieve messages with attachment URLs

5.4.3.3.1 Request

GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/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: 12345
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>MSISDN1</destinationAddress>
    <senderAddress>MSISDN2</senderAddress>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}</resourceURL>
    <messageId>{messageId1}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
      <link rel="attachment" href="http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}"

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document
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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients wants to use</td>
</tr>
<tr>
<td>registrationId</td>
<td>reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier</td>
</tr>
</tbody>
</table>
5.5.2  Response Codes

5.5.2.1  HTTP Response Codes
For HTTP response codes, see [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 ...[/apiVersion]/messaging/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages HTTP/1.1
Accept: application/xml
Content-Length: nnn
Content-Type: application/xml
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

```xml
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>MSISDN1</destinationAddress>
    <senderAddress>MSISDN2</senderAddress>
    <!-- resourceURL is not included because message is deleted from the server already -->
    <messageId>{messageId1}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
  <!-- MMS  -->
  <inboundMessage>
    <destinationAddress>MSISDN3</destinationAddress>
    <senderAddress>MSISDN4</senderAddress>
    <!-- resourceURL is not included because message is deleted from the server already -->
    <messageId>{messageId2}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
    </inboundMMSMessage>
  </inboundMessage>
</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients want to use</td>
</tr>
<tr>
<td>registrationId</td>
<td>reference to the retrieval criteria provisioned in advance and known to the client application. Analog of ParlayX registrationIdentifier</td>
</tr>
<tr>
<td>messageId</td>
<td>unique message identifier generated by server</td>
</tr>
</tbody>
</table>

5.6.2 Response Codes

5.6.2.1 HTTP Response Codes

For HTTP response codes, see [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 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.

5.6.4.1 Example: Read and delete one message (informative)

5.6.4.1.1 Request

```xml
POST ...(apiVersion)/messaging/inbound/registrations/{registrationId}/messages/{messageId}retrieveAndDelete HTTP/1.1
Accept: application/xml
Content-Length: nnn
Content-Type: application/xml
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>
```
5.6.4.1.2 Response

HTTP/1.1 200 OK
Content-Length: nnnnn
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>MSISDN1</destinationAddress>
  <senderAddress>MSISDN2</senderAddress>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}</resourceURL>
  <messageId>{messageId1}</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; charset=UTF-8
Content-Transfer-Encoding: 8 bit

Look at the attached picture

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

GIF89a...binary image data...

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

5.6.5 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].
5.6.6 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients want to use</td>
</tr>
<tr>
<td>registrationId</td>
<td>reference to the provisioned in advance and known to the client application. Analogous to Parlay X registrationIdentifier</td>
</tr>
<tr>
<td>messageId</td>
<td>unique message identifier generated by server</td>
</tr>
</tbody>
</table>

5.7.2 Response Codes

5.7.2.1 HTTP Response Codes

For HTTP response codes, see [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

```
GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId} ?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: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessage xmlns:mms="urn:oma:xml:rest:messaging:1">
  <destinationAddress>MSISDN1</destinationAddress>
  <senderAddress>MSISDN2</senderAddress>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}</resourceURL>
  <messageId>{messageId1}</messageId>
  <inboundMMSMessage>
    <subject>Who is RESTing on the beach?</subject>
  </inboundMMSMessage>
</mms:inboundMessage>

Look at the attached picture

Look at the attached picture

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 ../(apiVersion)/messaging/inbound/registrations/{registrationId}/messages/{messageId} 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
</tbody>
</table>
### 5.8.2 Response Codes

#### 5.8.2.1 HTTP Response Codes

For HTTP response codes, see [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)

**Request**

```
GET .../{'apiVersion'}/messaging/inbound/registrations/{'registrationId'}/messages/{'messageId'}/attachments/{'attachmentId'} HTTP/1.1
Accept: image/gif, image/png, image/jpeg, text/html, application/xml
Host: example.com:80
```

**Response**

```
HTTP/1.1 200 OK
Content-Length: nnnnnn
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.

---

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

5.8.6.1.1 Request

```
DELETE .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}/attachments/{attachmentId} 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients want to use (e.g. version 1 for 1.x)</td>
</tr>
</tbody>
</table>

5.9.2 Response Codes

5.9.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../{apiVersion}/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
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>680180999</destinationAddress>
    <criteria>Urgent*</criteria>
    <clientCorrelator>567891</clientCorrelator>
    <resourceURL>http://{serverRoot}/{apiVersion}/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>80999</destinationAddress>
    <criteria>Urgent*</criteria>
    <clientCorrelator>567892</clientCorrelator>
    <resourceURL>http://{serverRoot}/{apiVersion}/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

5.9.5.1.1 Request

POST .../{apiVersion}/messaging/inbound/subscriptions HTTP/1.1
Accept: application/xml
Content-Type: application/xml
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>+34680180999</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
Location: http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}
Content-Length: 254
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>
    <callbackData>12345</callbackData>
    <notificationFormat>XML</notificationFormat>
  </callbackReference>
  <destinationAddress>+34680180999</destinationAddress>
  <clientCorrelator>567893</clientCorrelator>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}</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 .../{apiVersion}/messaging/inbound/subscriptions HTTP/1.1
Accept: application/xml
Content-Type: application/xml
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>+34680180999</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://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}
Content-Length: 254
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://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)</td>
</tr>
<tr>
<td>subscriptionId</td>
<td>identifies the subscription</td>
</tr>
</tbody>
</table>

5.10.2 Response Codes

5.10.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../[apiVersion]/messaging/inbound/subscriptions/{subscriptionId} 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
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>+34680180999</destinationAddress>
  <criteria>Urgent*</criteria>
  <clientCorrelator>567893</clientCorrelator>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId}</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

5.10.6.1.1 Request

```
DELETE .../(apiVersion)/messaging/inbound/subscriptions/{subscriptionId} 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 [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

5.11.5.1.1 Request

```
POST .../notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/xml
Content-Length: 12345
Host: application.example.com:80
```
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 .../notifications/DeliveryInfoNotification HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Content-Length: 12345
Host: application.example.com:80
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<mms:inboundMessageNotification xmlns:mms="urn:oma:xml:rest:messaging:1">
  <inboundMessage>
    <destinationAddress>MSISDN1</destinationAddress>
    <senderAddress>MSISDN2</senderAddress>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}</resourceURL>
    <link rel="Subscription" href="http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}"/>
    <messageId>{messageId1}</messageId>
    <inboundMMSMessage>
      <subject>Who is RESTing on the beach?</subject>
      <link rel="attachment" href="http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}/attachments/{attachmentId1}"/>
      <link rel="attachment" href="http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}/attachments/{attachmentId2}"/>
    </inboundMMSMessage>
  </inboundMessage>
</mms:inboundMessageNotification>
<bodyText>Look at the attached picture</bodyText>
</inboundMMSMessage>
</inboundMessage>
</mms:inboundMessageNotification>

5.11.5.2.2 Response

HTTP/1.1 200 OK
Content-Type: application/xml
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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)</td>
</tr>
<tr>
<td>senderAddress</td>
<td>sender application address. Typically SHORT CODE [REST_TS_COMMON], but could be a terminal address</td>
</tr>
</tbody>
</table>

5.12.2 Response Codes

5.12.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../(apiVersion)/messaging/outbound/{senderAddress}/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
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:1350000001</address>
    <senderAddress>tel:1350000009</senderAddress>
    <outboundMMSMessage>
      <subject>Holiday greetings</subject>
    </outboundMMSMessage>
    <clientCorrelator>567894</clientCorrelator>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId1}</resourceURL>
    <deliveryInfoList>
      <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId1}/deliveryInfos</resourceURL>
      <deliveryInfo>
        <address>tel:1350000001</address>
        <deliveryStatus>DeliveredToTerminal</deliveryStatus>
      </deliveryInfo>
    </deliveryInfoList>
  </outboundMessageRequest>
</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 .../(apiVersion)/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name="root-fields"
Content-Type: application/xml; charset="utf-8"
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress>tel:1351111999</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
    <priority>High</priority>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
</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...
--===============123456==--

5.12.5.1.2 Response

HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document
5.12.5.2 Example 2: Create outgoing message, returning the location of created resource (Informative)

5.12.5.2.1 Request

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
<address>tel:1350000001</address>
<address>tel:1350000999</address>
<senderAddress>tel:1351111999</senderAddress>
<senderName>MyName</senderName>
<receiptRequest>
<notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</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;
5.12.5.2.2 Response

HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
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://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId} </resourceURL>
</common:resourceReference>

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

5.12.5.3.1 Request

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/related; boundary="123456=="; type="application/xml"
Host: example.com:80
MIME-Version: 1.0

--123456==
Content-Type: application/xml
<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress> tel:1351111999</senderAddress>
  <senderName>MyName</senderName>
  <charging>
    <description>Sample text for the charging information</description>
  </charging>
  <receiptRequest><!-- this is optional -->
    <notifyURL>http://example-application.com/notifications/DeliveryInfoNotification</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>hello from the rest of us!</subject>
  </outboundMMSMessage>
</mms:outboundMessageRequest>

See attached photo

---123456==
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif
GIF89a...binary image data...
---123456==
5.12.5.3.2 Response for charging not supported

HTTP/1.1 400 Bad request
Content-Type: application/xml
Content-Length: 254
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

5.12.5.4.1 Request

POST .../{apiVersion}/messaging/outbound/({senderAddress})/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

<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="picture.gif"
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...

--===============123456==--
5.12.5.4.2 Response

HTTP/1.1 400 Bad Request
Content-Type: application/xml
Content-Length: nnn
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  

5.12.5.5.1  Request

POST /apiVersion/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress>tel:1351111999</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</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="===123455=="

===123455===
Content-Disposition: attachments; filename="picture.gif"
Content-Type: text/plain;

See attached photo

===123455===
Content-Disposition: attachment; filename="picture.gif"
Content-Type: image/gif

GIF89a...binary image data...
--===============123456==--
5.12.5.5.2 Response

HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
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:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress>tel:1351111999</senderAddress>
  <senderName>MyName</senderName>
  <outboundMMSMessage>
    <subject> hello from the rest of us!</subject>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}</resourceURL>
  <deliveryInfoList>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/deliveryInfos</resourceURL>
    <deliveryInfo>
      <address>tel:+1350000001</address>
      <deliveryStatus>MessageWaiting</deliveryStatus>
    </deliveryInfo>
    <deliveryInfo>
      <address>tel:+1350000999</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 .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

<?xml version="1.0" encoding="UTF-8"?>
<mms:outboundMessageRequest xmlns:mms="urn:oma:xml:rest:messaging:1">
  <address>tel:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress>tel:1351111999</senderAddress>
</mms:outboundMessageRequest>
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
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
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:1350000001</address><address>tel:1350000999</address><senderAddress>tel:1351111999</senderAddress><senderName>MyName</senderName><outboundMMSMessage><subject> hello from the rest of us!</subject><priority>High</priority></outboundMMSMessage><clientCorrelator>567895</clientCorrelator></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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)</td>
</tr>
<tr>
<td>senderAddress</td>
<td>identifies client application. Typically SHORT CODE [REST_TS_COMMON], but could be a terminal address</td>
</tr>
<tr>
<td>requestId</td>
<td>outbound message request identifier generated by server</td>
</tr>
</tbody>
</table>

5.13.2 Response Codes

5.13.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId} 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: 12345
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:1350000001</address>
  <address>tel:1350000999</address>
  <senderAddress>tel:1351111999</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</notifyURL>
    <callbackData>12345</callbackData>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>Holiday greetings</subject>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}</resourceURL>
  <deliveryInfoList>
    <deliveryInfo>
      <address>tel:1350000001</address>
      <deliveryStatus>MessageWaiting</deliveryStatus>
    </deliveryInfo>
    <deliveryInfo>
      <address>tel:1350000999</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)</td>
</tr>
<tr>
<td>senderAddress</td>
<td>identifies client application. Typically SHORT CODE [REST_TS_COMMON], but could be a terminal address</td>
</tr>
<tr>
<td>requestId</td>
<td>outbound message request identifier generated by server</td>
</tr>
</tbody>
</table>

5.14.2 Response Codes

5.14.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../(apiVersion)/messaging/outbound/(senderAddress)/requests/(requestId)/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
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
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/inbound/{senderAddress}/subscriptions

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

5.15.1 Request URI variables

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API clients want to use</td>
</tr>
<tr>
<td>senderAddress</td>
<td>identifies client application. Typically SHORT CODE [REST_TS_COMMON], but could be a terminal address.</td>
</tr>
</tbody>
</table>
5.15.2 Response Codes

5.15.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../{apiVersion}/messaging/outbound/{senderAddress}/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
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://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions</resourceURL>
  <deliveryReceiptSubscription>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</notifyURL>
      <callbackData>12345</callbackData>
    </callbackReference>
    <filterCriteria>0102</filterCriteria>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}</resourceURL>
  </deliveryReceiptSubscription>
  <deliveryReceiptSubscription>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification</notifyURL>
      <callbackData>54321</callbackData>
    </callbackReference>
    <filterCriteria>0103</filterCriteria>
    <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}</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 .../(apiVersion)/messaging/outbound/{senderAddress}/subscriptions HTTP/1.1
Accept: application/xml
Content-Type: application/xml
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
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: ../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId}

<?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://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId}</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverRoot</td>
<td>server base url: hostname+port+base path. Example: <a href="http://example.com:80/ParlayREST">http://example.com:80/ParlayREST</a></td>
</tr>
<tr>
<td>apiVersion</td>
<td>version of the ParlayREST API client wants to use (e.g. version 1 for 1.x)</td>
</tr>
<tr>
<td>senderAddress</td>
<td>sender application address. Typically SHORT CODE [REST_TS_COMMON], but could be a terminal address.</td>
</tr>
<tr>
<td>subscriptionId</td>
<td>identifier of the subscription.</td>
</tr>
</tbody>
</table>

5.16.2 Response Codes

5.16.2.1 HTTP Response Codes

For HTTP response codes, see [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 .../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId} 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
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</notifyURL>
    <callbackData>12345</callbackData>
  </callbackReference>
  <filterCriteria>0102</filterCriteria>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}</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 ../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId} 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 [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 .../notifications)/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
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:1350000001</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
  </deliveryInfo>
  <deliveryInfo>
    <address>tel:1350000999</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
  </deliveryInfo>
  <link rel="OutboundMessageRequest" href="http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}" />
</mms:deliveryInfoNotification>
```
5.17.5.1.2  Response

HTTP/1.1 200 OK
Content-Type: application/xml
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

5.17.5.2.1  Request

POST .../notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
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:1350000999</address>
    <deliveryStatus>DeliveredToTerminal</deliveryStatus>
  </deliveryInfo>
  <link rel="OutboundMessageRequest" href="http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}"/>
</mms:deliveryInfoNotification>

5.17.5.2.2  Response

HTTP/1.1 200 OK
Content-Type: application/xml
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

A.1 Approved Version History

<table>
<thead>
<tr>
<th>Reference</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n/a</td>
<td>n/a</td>
<td>No prior version –or- No previous version within OMA</td>
</tr>
</tbody>
</table>

A.2 Draft/Candidate Version 1.0 History

<table>
<thead>
<tr>
<th>Document Identifier</th>
<th>Date</th>
<th>Sections</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>26 Nov 2009</td>
<td></td>
<td>Had to create a new version to upload the second part.</td>
</tr>
<tr>
<td>16 Dec 2009</td>
<td>All</td>
<td></td>
<td>Editorial fixes: Styles as per template. History Table.</td>
</tr>
<tr>
<td>26 Jan 2010</td>
<td>All</td>
<td></td>
<td>OMA-ARC-REST-2009-177.</td>
</tr>
<tr>
<td>Document Identifier</td>
<td>Date</td>
<td>Sections</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>05 Feb 2010</td>
<td>All</td>
<td>C001, C002, C006, C005, C022, C045, C036, C032, C039 C008, C009, C010, C011, C012, C013, C017, C018, C019, C020, C021, C022, C023, C024, C025, C026, C027, C028, C029, C033, C034, C035 C037, C038, C040, C041, C043, C044, C045, C046, C047, C048, C049 C050, C051, C052, C053, C054, C055, C056, C057, C058, C059, C060 C061, C062, C063, C064, C065, C066, C069, C070, C071, C072, C073 C074, C075, C077, C078, C079,</td>
<td></td>
</tr>
<tr>
<td>24 Feb 2010</td>
<td>Many</td>
<td>CRs implemented: OMA-ARC-REST-2009-0177 (fixed missing change in section 2.2) OMA-ARC-REST-2010-0037 (added missing references to SVC and POL exceptions) OMA-ARC-REST-2010-0042 (fixed missing change in section 2.2) OMA-ARC-REST-2010-0053R01 OMA-ARC-REST-2010-0008R03 OMA-ARC-REST-2010-0009R02 OMA-ARC-REST-2010-0010R02 (closing C0040, C0035, C0064) CONRR comments: C001 (completed inbound part for IM) C0019, B004 (SMS comment to remove “online”/”offline” is applicable to MMS also, agreed by e-mail mandate on Feb 24 2010). Other changes: “DeliveryInfoList” replaced with “deliveryInfos” in resources, as agreed during CONRR discussions. Subheadings for Example sections re-arranged as agreed during CONRR discussions.</td>
<td></td>
</tr>
<tr>
<td>26 Feb 2010</td>
<td>Many</td>
<td>All XML examples replaced with validated one. Also some small changes forgotten in the previous version.</td>
<td></td>
</tr>
<tr>
<td>09 Mar 2010</td>
<td>Many</td>
<td>CRs implemented: OMA-ARC-REST-2010-0110 OMA-ARC-REST-2010-0107R01 OMA-ARC-REST-2010-0090 OMA-ARC-REST-2010-0088R02 OMA-ARC-REST-2010-0085 OMA-ARC-REST-2010-0111R01-INP (the table in chapter 5.1 has been split and the header of the table is repeated for each page)</td>
<td></td>
</tr>
<tr>
<td>17 Mar 2010</td>
<td>Many</td>
<td>Implemented all CRs that were agreed in REST R&amp;A on March 10 OMA-ARC-REST-2010-0094 OMA-ARC-REST-2010-0102 OMA-ARC-REST-2010-0104R01 OMA-ARC-REST-2010-0106 Other CRs: OMA-ARC-REST-2010-74 Editorial fixes after a walk-through. <strong>Note that this revision contains changes that require update of XSDs</strong></td>
<td></td>
</tr>
<tr>
<td>30 Mar 2010</td>
<td>All</td>
<td>Implemented agreed CRs: OMA-ARC-REST-2010-0143R02 OMA-ARC-REST-2010-0150 OMA-ARC-REST-2010-0153 OMA-ARC-REST-2010-0156 Editorial fixes and 2010 copyright</td>
<td></td>
</tr>
<tr>
<td>Document Identifier</td>
<td>Date</td>
<td>Sections</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Candidate Version:</td>
<td>27 Apr 2010</td>
<td>All</td>
<td>Status changed to Candidate by TP: OMA-TP-2010-0186-INP_ParlayREST_V1_0_ERP_for_Candidate_Approval</td>
</tr>
<tr>
<td>OMA-TS-ParlayREST_MultiMediaMessaging-V1_0</td>
<td>09 Jun 2010</td>
<td>5.1, 5.2.11, 5.2.15, 5.2.19, 5.3.1, 5.5.5.1, 5.11.5, 5.12.5</td>
<td>Implemented agreed CRs: OMA-ARC-REST-2010-0196R01-CR_DataStructures_column_in_MMS OMA-ARC-REST-2010-0200-CR_Bugfix_moving_senderAddress_MMS OMA-ARC-REST-2010-0206-CR_binarySMS_in_MMS OMA-ARC-REST-2010-0256-CR_TS_MMS_Resolving_XML_validation_errors OMA-ARC-REST-2010-0267R01-CR_Bugfix_InboundMessageRetrieveAndDeleteRequest_TS_MMS</td>
</tr>
<tr>
<td>Draft Versions:</td>
<td>22 Jun 2010</td>
<td>C.3.1.2, C.4.1.2</td>
<td>Implemented agreed XML fix: OMA-ARC-REST-2010-0296-INP_XML_Errors_OMA_TS_ParlayREST_MultiMediaMessaging_V1_0_20100609_D</td>
</tr>
<tr>
<td>OMA-TS-ParlayREST_MultiMediaMessaging-V1_0</td>
<td>13 Jul 2010</td>
<td>5.4.3.2, B.17, D.2</td>
<td>Implemented agreed CRs: OMA-ARC-REST-2010-0313-CR_TS_SCR_changes_for_MMS OMA-ARC-REST-2010-0331-CR_Error_404_TS_MMS</td>
</tr>
<tr>
<td></td>
<td>14 Jul 2010</td>
<td>5.5.5.1.1, 5.15.3.1.2, 5.16.3.1.2</td>
<td>Editorial fixes to XML examples</td>
</tr>
<tr>
<td>Candidate Version:</td>
<td>24 Aug 2010</td>
<td>All</td>
<td>Status changes to Candidate by TP: OMA-TP-2010-0359-INP_ParlayREST_V1_0_ERP_for_Candidate_reapproval</td>
</tr>
<tr>
<td>OMA-TS-ParlayREST_MultiMediaMessaging-V1_0</td>
<td>08 Oct 2010</td>
<td>3.2</td>
<td>Fixed ref to [OMADICT]</td>
</tr>
<tr>
<td>Candidate Version:</td>
<td>23 Nov 2010</td>
<td>All</td>
<td>Status changed to Candidate by TP: OMA-TP-2010-0463R01-INP_ParlayREST_V1_0_ERP_for_Candidate_reapproval</td>
</tr>
</tbody>
</table>
Appendix B. Static Conformance Requirements (Normative)

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

## B.1 SCR for ParlayREST.MMS Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-SUPPORT-S-001-M</td>
<td>Support for the MMS REST Enabler</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>PARLAYREST-MMS-SUPPORT-S-002-M</td>
<td>Support for the XML request &amp; response format</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>PARLAYREST-MMS-SUPPORT-S-003-M</td>
<td>Support for the JSON request &amp; response format</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>PARLAYREST-MMS-SUPPORT-S-004-O</td>
<td>Support for the application/form-urlencoded format</td>
<td></td>
<td>Appendix C</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-INB-OFF-S-001-M</td>
<td>Support for reliable inbound messages delivery</td>
<td></td>
<td>5.4</td>
</tr>
<tr>
<td>PARLAYREST-MMS-INB-OFF-S-002-M</td>
<td>Retrieve messages from server - GET</td>
<td></td>
<td>5.4.3</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-INB-OFF-RETDEL-S-001-O</td>
<td>Support for inbound message delivery and delete</td>
<td></td>
<td>5.5</td>
</tr>
<tr>
<td>PARLAYREST-MMS-INB-OFF-RETDEL-S-002-O</td>
<td>Retrieve and delete messages from server - POST</td>
<td></td>
<td>5.5.5</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-MIME-INB-OFF-RETDEL-S-001-O</td>
<td>Support for inbound message delivery and delete</td>
<td></td>
<td>5.6</td>
</tr>
<tr>
<td>PARLAYREST-MMS-MIME-INB-OFF-RETDEL-S-002-O</td>
<td>Retrieve and delete one message from server - POST</td>
<td></td>
<td>5.6.4</td>
</tr>
</tbody>
</table>
### B.1.4 SCR for ParlayREST.MMS.Individual.Inbound Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-IND-INB-S-001-M</td>
<td>Support for inbound individual message delivery</td>
<td>5.7</td>
<td></td>
</tr>
<tr>
<td>PARLAYREST-MMS-IND-INB-S-002-O</td>
<td>Retrieve one message from server - GET</td>
<td>5.7.3</td>
<td></td>
</tr>
<tr>
<td>PARLAYREST-MMS-IND-INB-S-003-M</td>
<td>Confirm and delete retrieved message from server - DELETE</td>
<td>5.7.6</td>
<td></td>
</tr>
</tbody>
</table>

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

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

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

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

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-INB-INDON-SUBSCR-S-001-M</td>
<td>Support for control and read access to individual inbound subscription</td>
<td>5.10</td>
<td></td>
</tr>
<tr>
<td>Item</td>
<td>Function</td>
<td>Reference</td>
<td>Requirement</td>
</tr>
<tr>
<td>------</td>
<td>----------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>PARLAYREST-MMS-INB-INDON-SUBSCR-S-002-O</td>
<td>Read individual inbound subscription - GET</td>
<td>5.10.3</td>
<td></td>
</tr>
<tr>
<td>PARLAYREST-MMS-INB-INDON-SUBSCR-S-003-M</td>
<td>Update individual inbound subscriptions - DELETE</td>
<td>5.10.6</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-INB-NOTIF-S-001-M</td>
<td>Support for notifying application about inbound messages</td>
<td>5.11</td>
<td></td>
</tr>
<tr>
<td>PARLAYREST-MMS-INB-NOTIF-S-002-M</td>
<td>Notify application about inbound message arrival - POST</td>
<td>5.11.5</td>
<td></td>
</tr>
</tbody>
</table>

**B.1.9 SCR for ParlayREST.MMS.Outbound Server**

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

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-OUTB-MSGDELSTAT-S-002-O</td>
<td>Retrieve Outgoing Message Delivery Status - GET</td>
<td>5.13.3</td>
<td></td>
</tr>
</tbody>
</table>
### B.1.11 SCR for ParlayREST.MMS.Outbound.DeliveryStatus Server

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-OUTB-DELSSTAT-S-001-M</td>
<td>Support for requesting delivery status of outbound messages</td>
<td>5.14</td>
<td></td>
</tr>
</tbody>
</table>

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

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

### B.1.13 SCR for ParlayREST.MMS.Individual.Outbound.Subscript Server

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

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

<table>
<thead>
<tr>
<th>Item</th>
<th>Function</th>
<th>Reference</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARLAYREST-MMS-OUTB-DELSNOTIF-S-001-M</td>
<td>Support for notifying application about delivery status of outbound messages</td>
<td>5.17</td>
<td></td>
</tr>
<tr>
<td>PARLAYREST-MMS-OUTB-DELSNOTIF-S-002-M</td>
<td>Notify application about delivery status of outbound message - POST</td>
<td>5.17.5</td>
<td></td>
</tr>
</tbody>
</table>
Appendix C. Application/x-www-form-urlencoded Request Format for Selected Operations

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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>xsd:anyURI [1...unbounded]</td>
<td>No</td>
<td>Destination address(es) for the message</td>
</tr>
<tr>
<td>senderAddress</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>The address of the sender to whom a responding SMS may be sent. If the address is in the form of an MSISDN, include the protocol prefix 'tel:' and '%2B' followed by the country code before the subscriber number; e.g. tel:%2B447990123456.</td>
</tr>
<tr>
<td>senderName</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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).</td>
</tr>
<tr>
<td>chargingDescription</td>
<td>xsd:string [0..unbounded]</td>
<td>Yes</td>
<td>Description of charge to apply to this message. In case charging is required, this parameter MUST be present.</td>
</tr>
<tr>
<td>chargingCurrency</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Currency of charge to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.</td>
</tr>
<tr>
<td>chargingAmount</td>
<td>xsd:decimal</td>
<td>Yes</td>
<td>Charging amount to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.</td>
</tr>
<tr>
<td>chargingCode</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Charging code to apply to this message. In case chargingDescription is not present, this parameter MUST NOT be present.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>notifyURL</td>
<td>xsd:anyURI</td>
<td>Yes</td>
<td>URL to notify the application for delivery receipts. MUST NOT be present.</td>
</tr>
<tr>
<td>callbackData</td>
<td>xsd:string</td>
<td>Yes</td>
<td>Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.</td>
</tr>
<tr>
<td>notificationFormat</td>
<td>common:NotificationFormat</td>
<td>Yes</td>
<td>Default: XML. Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.</td>
</tr>
<tr>
<td>clientCorrelator</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>subject</td>
<td>xsd:string</td>
<td>Yes</td>
<td>If present, indicates the subject of the received message.</td>
</tr>
<tr>
<td>priority</td>
<td>MessagePriority</td>
<td>Yes</td>
<td>The priority of the message: default is Normal.</td>
</tr>
</tbody>
</table>

C.1.1 Example: Create outgoing message (Informative)

C.1.1.1 Request

```
POST .../(apiVersion)/messaging/outbound/(senderAddress)/requests HTTP/1.1
Content-Length: nnnnnn
Content-Type: multipart/form-data;
    boundary="===============123456==";
MIME-Version: 1.0
Host: www.example.com
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:+13500000991&
    address=tel:+13500000992&
    senderAddress=tel:+12345678&
    Subject=My%20message&
    notifyURL=http://example-application.com/notifications/DeliveryInfoNotification/54311&
    clientCorrelator=123456&senderName=Bob
```
C.1.1.2 Response

HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 12345
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:135000000991</address>
  <address>tel:13500000992</address>
  <senderAddress>tel:12345678</senderAddress>
  <senderName>MyName</senderName>
  <receiptRequest>
    <notifyURL>http://application.example.com/notifications/DeliveryInfoNotification/54311</notifyURL>
  </receiptRequest>
  <outboundMMSMessage>
    <subject>Holiday greetings</subject>
  </outboundMMSMessage>
  <clientCorrelator>567895</clientCorrelator>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterCriteria</td>
<td>xsd:string</td>
<td>No</td>
<td>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.</td>
</tr>
<tr>
<td>notifyURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Notification endpoint definition</td>
</tr>
<tr>
<td>callbackData</td>
<td>xsd:string</td>
<td>No</td>
<td>Data the application can register with the server when subscribing to notifications, and that are</td>
</tr>
</tbody>
</table>
C.2.1 Example: Create outbound delivery notification subscription

C.2.1.1 Request

POST ../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Accept: application/xml

filterCriteria=13500&
correlator=123456&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/12345

C.2.1.2 Response

HTTP/1.1 201 Created
Content-Type: application/xml
Date: Thu, 04 Jun 2009 02:51:59 GMT
Location: ../{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}

<?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://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId}</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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type/Values</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destinationAddress</td>
<td>xsd:anyURI[1..unbounded]</td>
<td>No</td>
<td>Destination address of the message</td>
</tr>
<tr>
<td>criteria</td>
<td>xsd:string</td>
<td>Yes</td>
<td>The text to match against to determine the application to receive the notification</td>
</tr>
<tr>
<td>notifyURL</td>
<td>xsd:anyURI</td>
<td>No</td>
<td>Notification endpoint definition</td>
</tr>
<tr>
<td>callbackData</td>
<td>xsd:string</td>
<td>No</td>
<td>Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.</td>
</tr>
<tr>
<td>notificationFormat</td>
<td>common:NotificationFormat</td>
<td>Yes</td>
<td>Default: XML&lt;br&gt;Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.</td>
</tr>
<tr>
<td>clientCorrelator</td>
<td>xsd:string</td>
<td>Yes</td>
<td>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.</td>
</tr>
<tr>
<td>useAttachmentURLs</td>
<td>xsd:boolean</td>
<td>Yes</td>
<td>Default: false&lt;br&gt;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</td>
</tr>
</tbody>
</table>

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 .../(apiVersion)/messaging/inbound/subscriptions HTTP/1.1
Host: www.example.com:80
Content-Type: application/x-www-form-urlencoded
Accept: application/xml
```
destinationAddress=81771&
criteria=Vote&
notifyURL=http://application.example.com/notifications/DeliveryInfoNotification/3455&
notificationFormat=XML

C.3.1.2  Response

HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}
Content-Length: 254
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>+34680180999</destinationAddress>
  <criteria>Urgent*</criteria>
  <resourceURL>http://{serverRoot}/{apiVersion}/messaging/inbound/subscription/{subscriptionId1}</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 [REST_TS_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [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:

```plaintext
GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageList": [
  "inboundMessage": [
    {
      "destinationAddress": "MSISDN1",
      "messageId": "{messageId1}",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId1}",
      "senderAddress": "MSISDN2"
    },
    {
      "destinationAddress": "MSISDN3",
      "messageId": "{messageId2}",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId2}",
      "senderAddress": "MSISDN4"
    }
  ],
  "numberOfMessagesInThisBatch": "2",
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages?maxBatchSize=2",
  "totalNumberOfPendingMessages": "20"
]}
```

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

Request:

```plaintext
GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages?maxBatchSize=2 HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageList": [
  "inboundMessage": [
    {
      "destinationAddress": "MSISDN1",
      "messageId": "{messageId1}",
      "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
      "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId1}",
      "senderAddress": "MSISDN2"
    }
  ],
  "numberOfMessagesInThisBatch": "2",
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages?maxBatchSize=2",
  "totalNumberOfPendingMessages": "20"
]}
```
D.3 Retrieve messages with attachment URLs (section 5.4.3.3)

Request:

GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/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: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageList": {
  "inboundMessage": {
    "destinationAddress": "MSISDN1",
    "messageId": "{messageId1}",
    "inboundMMSMessage": {
      "bodyText": "See attached picture",
      "link": [
        {
          "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId1}/attachments/{attachmentId1}"
        },
        {
          "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId1}/attachments/{attachmentId2}"  
        }
      ],
   "serviceException": {
      "messageId": "SVC0002",
      "text": "Invalid input value. The requested registration id: %1 does not exist",
      "variables": "registration123"
    }
  }
}}
D.4 Retrieve and delete inbound messages (section 5.5.5.1)

Request:

POST .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/retrieveAndDeleteMessages HTTP/1.1
Accept: application/json
Content-Length: nnn
Content-Type: application/json; charset=UTF-8
Host: example.com:80

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

Response:

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageList": {
    "inboundMessage": [
        {
            "destinationAddress": "MSISDN1",
            "messageId": "{messageId1}",
            "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
            "senderAddress": "MSISDN2"
        },
        {
            "destinationAddress": "MSISDN3",
            "messageId": "{messageId2}",
            "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
            "senderAddress": "MSISDN4"
        }
    ],
    "numberOfMessagesInThisBatch": "2",
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/retrieveAndDeleteMessages",
    "totalNumberOfPendingMessages": "20"}}
D.5 Read and delete one message (section 5.6.4.1)

Request:

POST .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}/retrieveAndDelete HTTP/1.1
Accept: application/json
Content-Length: nnn
Content-Type: application/json; charset=UTF-8
Host: example.com:80

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

Response:

HTTP/1.1 200 OK
Content-Length: nnnnnn
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": "MSISDN1",
  "messageId": "{messageId1}"
  "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/{messageId1}"
  "senderAddress": "MSISDN2"
}}

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

Request:

GET .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}?resFormat=JSON HTTP/1.1
Host: example.com:80

Response:

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

=====12345====
D.7 Remove message from gateway storage (section 5.7.6.1)

Request:

```
DELETE .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId} 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 .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}/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-Length: nnnnnn
Content-Type: image/gif
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 .../{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId}/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 .../{apiVersion}/messaging/inbound/subscriptions HTTP/1.1
Accept: application/json
Host: example.com:80

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

{"subscriptionList": ["subscription": [{
    "callbackReference": {
      "callbackData": "12345",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/12345"
    },
    "clientCorrelator": "567891",
    "criteria": "Urgent*",
    "destinationAddress": "680180999",
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/0000001",
    "useAttachmentURLs": "false"
  },
  {
    "callbackReference": {
      "callbackData": "54321",
      "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/54321"
    },
    "clientCorrelator": "567892",
    "criteria": "Urgent*",
    "destinationAddress": "80999",
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/0000002",
    "useAttachmentURLs": "false"
  }]}]}
D.11 Create inbound subscription (returning a representation of created resource) (section 5.9.5.1)

Request:

```
POST .../{apiVersion}/messaging/inbound/subscriptions HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=UTF-8
Host: example.com:80

{"subscription": {
    "callbackReference": {
        "callbackData": "12345",
        "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification/88888"
    },
    "clientCorrelator": "567893",
    "criteria": "Urgent***",
    "destinationAddress": "+34680180999",
    "useAttachmentURLs": "false"
}}
```

Response:

```text
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}
Content-Length: 254
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": "+34680180999",
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions{subscriptionId1}",
    "useAttachmentURLs": "false"
}}
```

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

Request:

```text
POST .../{apiVersion}/messaging/inbound/subscriptions HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=UTF-8
Host: example.com:80

{"subscription": {
}}
```
"callbackReference": {  
  "callbackData": "12345",  
  "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
},  
  "criteria": "Urgent"",  
  "destinationAddress": "+34680180999",  
  "useAttachmentURLs": "false"
}}

Response:

HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"resourceReference": {"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}"}}

D.13 Read individual subscription (section 5.10.3.1)

Request:

GET .../{apiVersion}/messaging/inbound/subscriptions/{subscriptionId} HTTP/1.1
Accept: application/json
Host: example.com:80

Response:

HTTP/1.1 200 OK
Content-Type: application/json
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": "+34680180999",  
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId}"  
  "useAttachmentURLs": "false"
  }  
}

D.14 Delete a subscription (section 5.10.6.1)

Request:

DELETE .../{apiVersion}/messaging/inbound/subscriptions/{subscriptionId} HTTP/1.1
D.15 Message arrival notification (section 5.11.5.1)

Request:

POST .../notifications/DeliveryInfoNotification/88888 HTTP/1.1
Accept: application/json
Content-Length: 12345
Host: application.example.com:80
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageNotification": {"inboundMessage": {
 "destinationAddress": "MSISDN1",
 "messageId": "{messageId1}",
 "inboundMMSMessage": {"subject": "Who is RESTing on the beach?"},
 "link": {
 "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}",
 "rel": "Subscription"
 },
 "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}",
 "senderAddress": "MSISDN2"
 }}}

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 http://example-application.com/notifications/DeliveryInfoNotification HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: 12345
Host: application.example.com:80
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"inboundMessageNotification": {"inboundMessage": {
 "destinationAddress": "MSISDN1",
 "messageId": "{messageId1}"
 }}}}
"messageId": "(messageId1)",
"inboundMMSMessage": {
  "bodyText": "Look at the attached picture",
  "link": [
    {
      "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/(messageId1)/attachments/(attachmentId1)",
      "rel": "attachment"
    },
    {
      "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/(messageId1)/attachments/(attachmentId2)",
      "rel": "attachment"
    }
  ],
  "subject": "Who is RESTing on the beach?"
},
"link": {
  "href": "http://{serverRoot}/{apiVersion}/messaging/inbound/subscriptions/{subscriptionId1}"
},
"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/inbound/registrations/{registrationId}/messages/{messageId1}"
"senderAddress": "MSISDN2"
}}

Response:

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

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

Request:

GET .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Accept: application/json
Host: example.com:80

Response:

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

{"outboundMessageRequestList": {"outboundMessageRequest": {
  "address": "tel:1350000001",
  "clientCorrelator": "567894",
  "deliveryInfoList": {
    "deliveryInfo": {
      "address": "tel:1350000001",
      "deliveryStatus": "DeliveredToTerminal"
    }
  }
}}
"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId1}/deliveryInfos"
},
"outboundMMSMessage": {"subject": "Holiday greetings"},
"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId1}"
"senderAddress": "tel:1350000009"
}}

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

Request:

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name="root-fields"
Content-Type: application/json; charset="utf-8"
Content-Length: nnnn

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ],
  "clientCorrelator": "567895",
  "outboundMMSMessage": {
    "priority": "High",
    "subject": "hello from the rest of us!"
  },
  "receiptRequest": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
  },
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}

Response:

HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ]}
D.19 Create outgoing message, returning the location of created resource (section 5.12.5.2)

Request:

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456=="
Host: example.com:80
MIME-Version: 1.0

--===============123456==
Content-Disposition: multipart/form-data; name="root-fields"
Content-Type: application/json; charset="utf-8"
Content-Length: nnnn

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ],
  "clientCorrelator": "567895",
  "outboundMMSMessage": {
    "priority": "High",
    "subject": "hello from the rest of us!"
  },
  "receiptRequest": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
  },
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}

Response:

HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"resourceReference": {"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}"}}
D.20 Create outgoing message with charging (section 5.12.5.3)

Request:

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/related; boundary="===============123456=="; type="application/json"
Host: example.com:80
MIME-Version: 1.0

--===============123456==
Content-Type: application/json; charset="utf-8"

{"outboundMessageRequest": {
    "address": [
        "tel:1350000001",
        "tel:1350000999"
    ],
    "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"
    },
    "senderAddress": "tel:1351111999",
    "senderName": "MyName"
}}

Response for charging not supported:

HTTP/1.1 400 Bad request
Content-Type: application/json
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"requestError": {"policyException": {
    "messageId": "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 .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0
D.22 Create outgoing message, multiple addresses partial success, with deliveryInfoList in response (section 5.12.5.5)

Request:

```
POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

{"outboundMessageRequest": {
    "address": [
        "tel:1350000001",
        "tel:1350000999"
    ],
    "clientCorrelator": "567895",
    "outboundMMSMessage": {
        "priority": "High",
        "subject": "hello from the rest of us!"
    },
    "receiptRequest": {
        "callbackData": "12345",
        "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
    },
    "senderAddress": "tel:1351111999",
    "senderName": "MyName"
}}
```
"address": [
  "tel:1350000001",
  "tel:1350000999"
],
"clientCorrelator": "567895",
"outboundMMSMessage": {
  "priority": "High",
  "subject": "hello from the rest of us!"
},
"receiptRequest": {
  "callbackData": "12345",
  "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
},
"senderAddress": "tel:1351111999",
"senderName": "MyName"
}

Response:

HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ],
  "clientCorrelator": "567895",
  "deliveryInfoList": {
    "deliveryInfo": [
      {
        "address": "tel:+1350000001",
        "deliveryStatus": "MessageWaiting"
      },
      {
        "address": "tel:+1350000999",
        "deliveryStatus": "DeliveryImpossible"
      }
    ],
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/deliveryInfos"
  },
  "outboundMMSMessage": {
    "subject": "hello from the rest of us!"
  },
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}",
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}
D.23 Create outgoing message, multiple addresses partial success, without deliveryInfoList in response (section 5.12.5.6)

Request:

POST .../{apiVersion}/messaging/outbound/{senderAddress}/requests HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com:80
MIME-Version: 1.0

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

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ],
  "clientCorrelator": "567895",
  "outboundMMSMessage": {
    "priority": "High",
    "subject": "hello from the rest of us!"
  },
  "receiptRequest": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
  },
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}

Response:

HTTP/1.1 201 Created
Content-Type: application/json
Location: http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}
Content-Length: 254
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:1350000999"
  ],
  "clientCorrelator": "567895",
  "outboundMMSMessage": {
    "subject": "hello from the rest of us!"
  },
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}",
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}
D.24 Read message request and delivery status (section 5.13.3.1)

Request:

```
GET .../{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId} HTTP/1.1
Accept: application/json
Host: example.com:80
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 12345
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"outboundMessageRequest": {
  "address": [
    "tel:1350000001",
    "tel:135000999"
  ],
  "clientCorrelator": "567895",
  "deliveryInfoList": {
    "deliveryInfo": [
    {
      "address": "tel:1350000001",
      "deliveryStatus": "MessageWaiting"
    },
    {
      "address": "tel:135000999",
      "deliveryStatus": "MessageWaiting"
    }
    ],
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/DeliveryInfos"
  },
  "outboundMMSMessage": {"subject": "Holiday greetings"},
  "receiptRequest": {
    "callbackData": "12345",
    "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
  },
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/",
  "senderAddress": "tel:1351111999",
  "senderName": "MyName"
}}
```

D.25 Read message delivery status (section 5.14.3.1)

Request:

```
GET .../{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/deliveryInfos HTTP/1.1
Accept: application/json
Host: example.com:80
```
Response:

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

{"deliveryInfoList": {
    "deliveryInfo": [
        {
            "address": "tel:1350000001",
            "deliveryStatus": "MessageWaiting"
        },
        {
            "address": "tel:1350000999",
            "deliveryStatus": "MessageWaiting"
        }
    ],
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/DeliveryInfo"
}}

D.26 Read delivery notification subscriptions (section 5.15.3.1)

Request:

GET ../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions HTTP/1.1
Accept: application/json
Host: example.com:80

Response:

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

{"deliveryReceiptSubscriptionList": {
    "deliveryReceiptSubscription": [
        {
            "callbackReference": {
                "callbackData": "12345",
                "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
            },
            "filterCriteria": "0102",
            "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}"
        },
        {
            "callbackReference": {
                "callbackData": "54321",
                "notifyURL": "http://application.example.com/notifications/DeliveryInfoNotification"
            },
            "filterCriteria": "0103",
            "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions/{subscriptionId}"
        }
    ],
    "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/subscriptions"
D.27  Create outbound delivery notification subscription (section 5.15.5.1)

Request:

POST .../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=UTF-8
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
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://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId}"}}

D.28  Read individual message delivery notification subscription (section 5.16.3.1)

Request:

GET .../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId} HTTP/1.1
Accept: application/json
Host: example.com:80

Response:

HTTP/1.1 200 OK
Content-Type: application/json
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://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/subscriptions/{subscriptionId}"}}
D.29 Delete message delivery notification subscription (section 5.16.6.1)

Request:

DELETE .../{apiVersion}/messaging/outbound/{senderAddress}/subscriptions /{subscriptionId} 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.30 Notify client about outbound message delivery status, multiple delivery status per notification (section 5.17.5.1)

Request:

POST http://example-application.com/notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=UTF-8
Host: application.example.com:80

{"deliveryInfoList": {
"deliveryInfo": [
  {
    "address": "tel:1350000001",
    "deliveryStatus": "DeliveredToTerminal"
  },
  {
    "address": "tel:1350000999",
    "deliveryStatus": "DeliveredToTerminal"
  }
],
"link": {
"href": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}"
,"rel": "OutboundMessageRequest"
}}
"resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/DeliveryInfos"
}}

Response:

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

D.31 Notify client about outbound message delivery status, single delivery status per notification (section 5.17.5.2)

Request:

POST .../notifications/DeliveryInfoNotification/77777 HTTP/1.1
Accept: application/json
Content-Type: application/json; charset=UTF-8
Host: example.com:80

{"deliveryInfoList": {
  "deliveryInfo": {
    "address": "tel:13500009999",
    "deliveryStatus": "DeliveredToTerminal"
  },
  "link": {
    "href": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}"
  },
  "resourceURL": "http://{serverRoot}/{apiVersion}/messaging/outbound/{senderAddress}/requests/{requestId}/DeliveryInfos"
}}

Response:

HTTP/1.1 200 OK
Content-Type: application/json
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.

<table>
<thead>
<tr>
<th>ParlayREST Resource</th>
<th>ParlayREST Method</th>
<th>ParlayREST Section reference</th>
<th>Parlay X equivalent operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbound messages for a given registration</td>
<td>GET</td>
<td>5.4.3</td>
<td>getReceivedMessages 1)</td>
</tr>
<tr>
<td>Inbound messages retrieve and delete using registration</td>
<td>POST</td>
<td>5.5.5</td>
<td>getReceivedMessages</td>
</tr>
<tr>
<td>Retrieval and deletion of individual inbound message using registration</td>
<td>POST</td>
<td>5.6.4</td>
<td>getMessage</td>
</tr>
<tr>
<td>Inbound message for a given registration</td>
<td>GET</td>
<td>5.7.3</td>
<td>getMessage, getMessageURIs 2)</td>
</tr>
<tr>
<td>Inbound message subscriptions</td>
<td>POST</td>
<td>5.9.5</td>
<td>startMessageNotification</td>
</tr>
<tr>
<td>Individual inbound message subscription</td>
<td>DELETE</td>
<td>5.10.6</td>
<td>stopMessageNotification</td>
</tr>
<tr>
<td>Client notification about inbound message</td>
<td>POST</td>
<td>5.11.5</td>
<td>notifyMessageReception</td>
</tr>
<tr>
<td>Outbound message requests</td>
<td>POST</td>
<td>5.12.5</td>
<td>sendMessage</td>
</tr>
<tr>
<td>Outbound message delivery status</td>
<td>GET</td>
<td>5.14.3</td>
<td>getMessageDeliveryStatus</td>
</tr>
<tr>
<td>Outbound message delivery notification subscriptions</td>
<td>POST</td>
<td>5.15.5</td>
<td>startDeliveryReceiptNotification</td>
</tr>
<tr>
<td>Individual outbound message delivery notification subscription</td>
<td>DELETE</td>
<td>5.16.6</td>
<td>stopDeliveryReceiptNotification</td>
</tr>
<tr>
<td>Client notification about outbound message delivery status</td>
<td>POST</td>
<td>5.17.5</td>
<td>notifyMessageDeliveryReceipt</td>
</tr>
</tbody>
</table>

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.