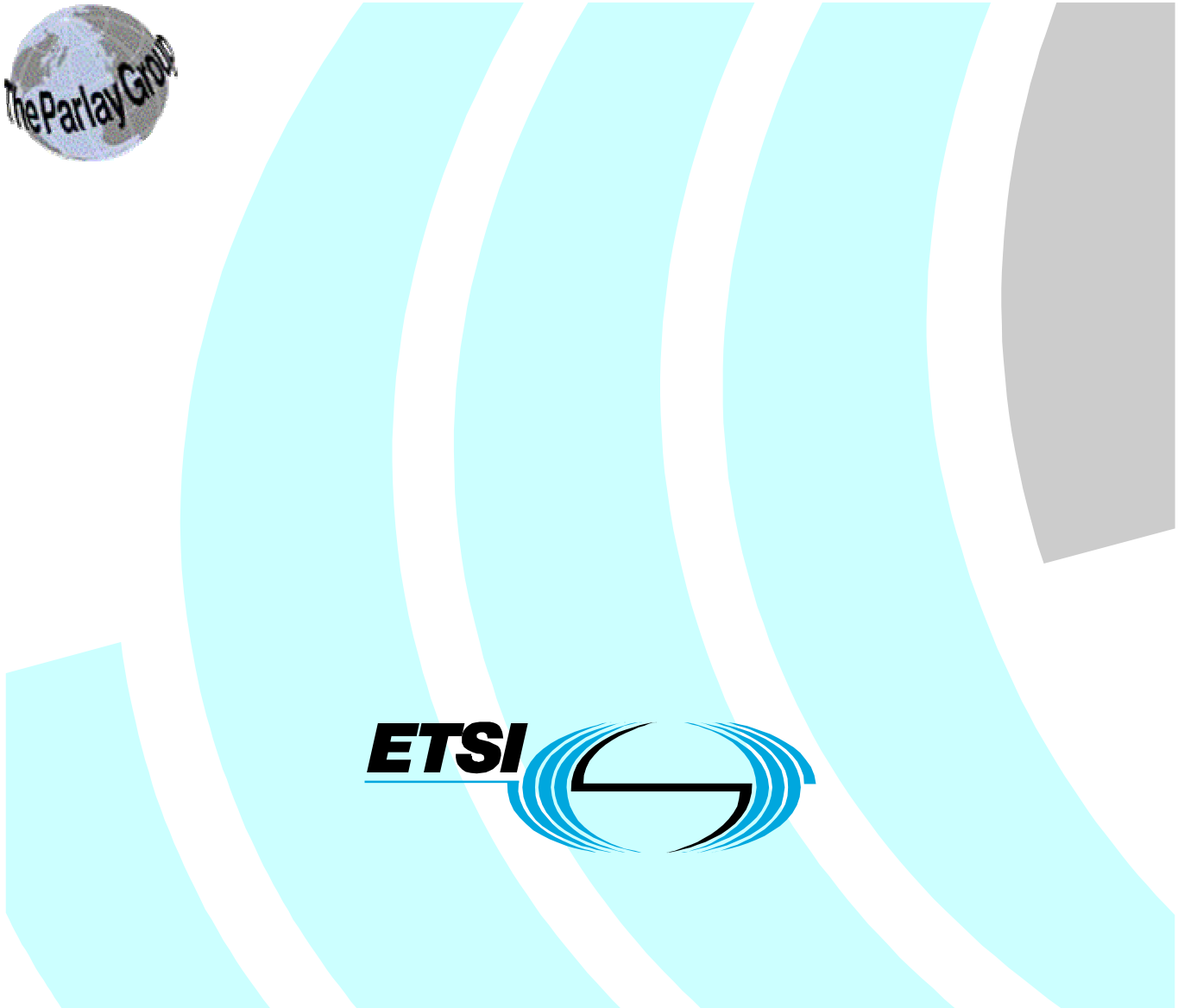
Final draft ETSI ES 202 391-5 V1.1.1 (2005-01)

ETSI Standard

Open Service Access (OSA);
Parlay X Web Services;
Part 5: Multimedia Messaging



Reference
DES/TISPAN-01007-05-OSA

Keywords
API, OSA, service

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document is part 5 of a multi-part deliverable covering Open Service Access (OSA); Parlay X Web Services, as identified below:

Part 1: "Common";

Part 2: "Third Party Call";

Part 3: "Call Notification";

Part 4: "Short Messaging";

Part 5: "Multimedia Messaging";

Part 6: "Payment";

Part 7: "Account Management";

Part 8: "Terminal Status";

Part 9: "Terminal Location";

Part 10: "Call Handling";

Part 11: "Audio Call";

Part 12: "Multimedia Conference";

Part 13: "Address List Management";

Part 14: "Presence".

The present document has been defined jointly between ETSI, The Parlay Group (http://www.parlay.org) and the 3GPP.

The present document is equivalent to 3GPP TS 29.199-05 V6.0.0 (Release 6).

1 Scope

The present document is part 5 of the Stage 3 Parlay X Web Services specification for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardized interface, i.e. the OSA APIs.

The present document specifies the Multimedia Messaging Web Service. The following are defined here:

- Name spaces.
- Sequence diagrams.
- Data definitions.
- Interface specification plus detailed method descriptions.
- Fault definitions.
- Service Policies.
- WSDL Description of the interfaces.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] W3C Recommendation (2 May 2001): "XML Schema Part 2: Datatypes".

NOTE: Available at http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.

[2] ETSI ES 202 391-1: "Open Service Access (OSA); Parlay X Web Services; Part 1: Common".

[3] W3C Note (11 December 2000): "SOAP Messages with Attachments".

NOTE: Available at http://www.w3.org/TR/SOAP-attachments.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 391-1 [2] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in ES 202 391-1 [2] and the following apply:

EMS	Enhanced Messaging Service
IM	Instant Messaging
MMS	Multimedia Messaging Service
MMS-C	Multimedia Messaging Service - Centre
SMS	Short Message Service

4 Detailed service description

Currently, in order to programmatically receive and send Multimedia Messages, it is necessary to write applications using specific protocols to access MMS functions provided by network elements (e.g. MMS-C). This approach requires application developers to have a high degree of network expertise.

This contribution defines a Multimedia Messaging Web Service that can map to SMS, EMS, MMS, IM, E-mail, etc.

The choice is between defining one set of interfaces per messaging network or a single set common to all networks; e.g. we could define sendMMS, sendEMS, sendSMS, etc., or just use sendMessage. Although the more specific the API the easier it is to use, there are advantages to a single set of network-neutral APIs. These advantages include:

- improved service portability;
- lower complexity, by providing support for generic user terminal capabilities only.

For this version of the Parlay X specification, we provide sets of interfaces for two messaging Web Services: Short Messaging (part 4) and Multimedia Messaging (this part), which provides generic messaging features (including SMS).

For sending a message to the network (see Send Message), the application invokes a message to send it and must subsequently become active again to poll for delivery status. There is an alternative to this polling mechanism, i.e. an asynchronous notification mechanism implemented with an application-side Web Service. However it was decided not to provide a notification mechanism in the first release, to make the interface as simple as possible, even though the polling mechanism is not as network efficient as the notification mechanism.

For receiving a message from the network, the application may use either polling (see Receive Message) or notification (see Message Notification) mechanisms. The notification mechanism is more common: network-initiated messages are sent to autonomous application-side Web Services. Both mechanisms are supported, but the provisioning of the notification-related criteria is not specified.

Figure 1 shows an example scenario using sendMessage and getMessageDeliveryStatus to send data to subscribers and to determine if the data has been received by the subscriber. The application invokes a Web Service to retrieve a stock quote (1) and (2) and sends the current quote - sendMessage - using the Parlay X Interface (3) of the Multimedia Messaging Web Service. After invocation, the Multimedia Message Web Service sends the message to an MMS-C using the MM7 interface (4) for onward transmission (5) to the subscriber over the Mobile network.

Later, when the next quote is ready, the application checks to see - getMessageDeliveryStatus - if the previous quote has been successfully delivered to the subscriber. If not, it may for instance perform an action (not shown) to provide a credit for the previous message transmission. This way, the subscriber is only charged for a stock quote if it is delivered on time.

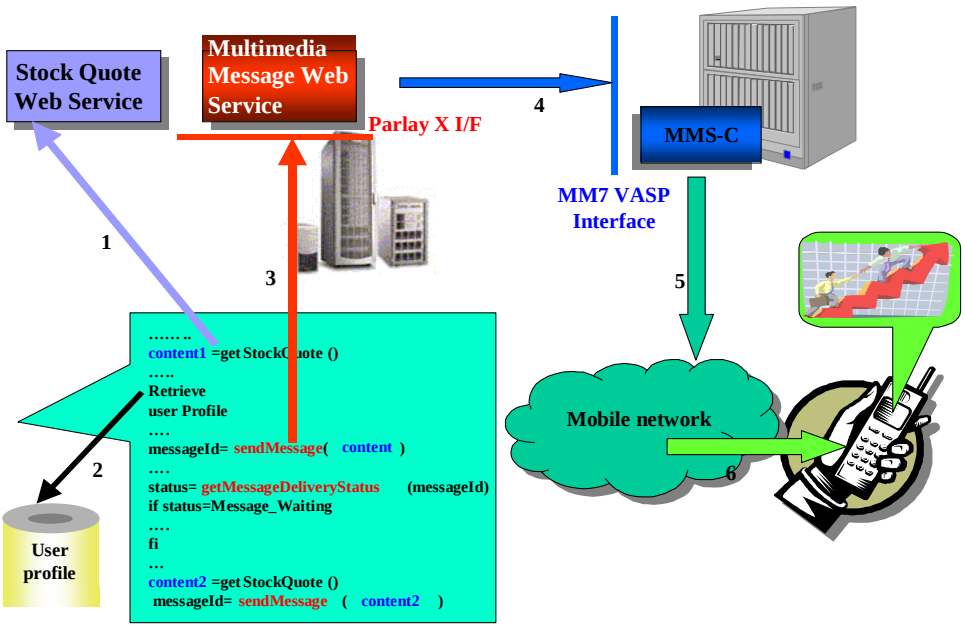


Figure 1: Multimedia Messaging Scenario

5 Namespaces

The SendMessage interface uses the namespace:

www.csapi.org/wsdl/parlayx/multimedia_messaging/send/v2_0

The ReceiveMessage interface uses the namespace:

www.csapi.org/wsdl/parlayx/multimedia_messaging/receive/v2_0

The MessageNotification interface uses the namespace:

www.csapi.org/wsdl/parlayx/multimedia_messaging/notification/v2_0

The data types are defined in the namespace:

www.csapi.org/schema/parlayx/multimedia_messaging/v2_0

The 'xsd' namespace is used in the present document to refer to the XML Schema data types defined in XML Schema [1]. The use of the name 'xsd' is not semantically significant.

6 Sequence diagrams

6.1 Send picture

With the advent of picture capable phones, the exchange of photos to mobile phones is becoming more common place. This sequence diagram shows an application where a person can send a picture from an online photo album to a mobile phone.

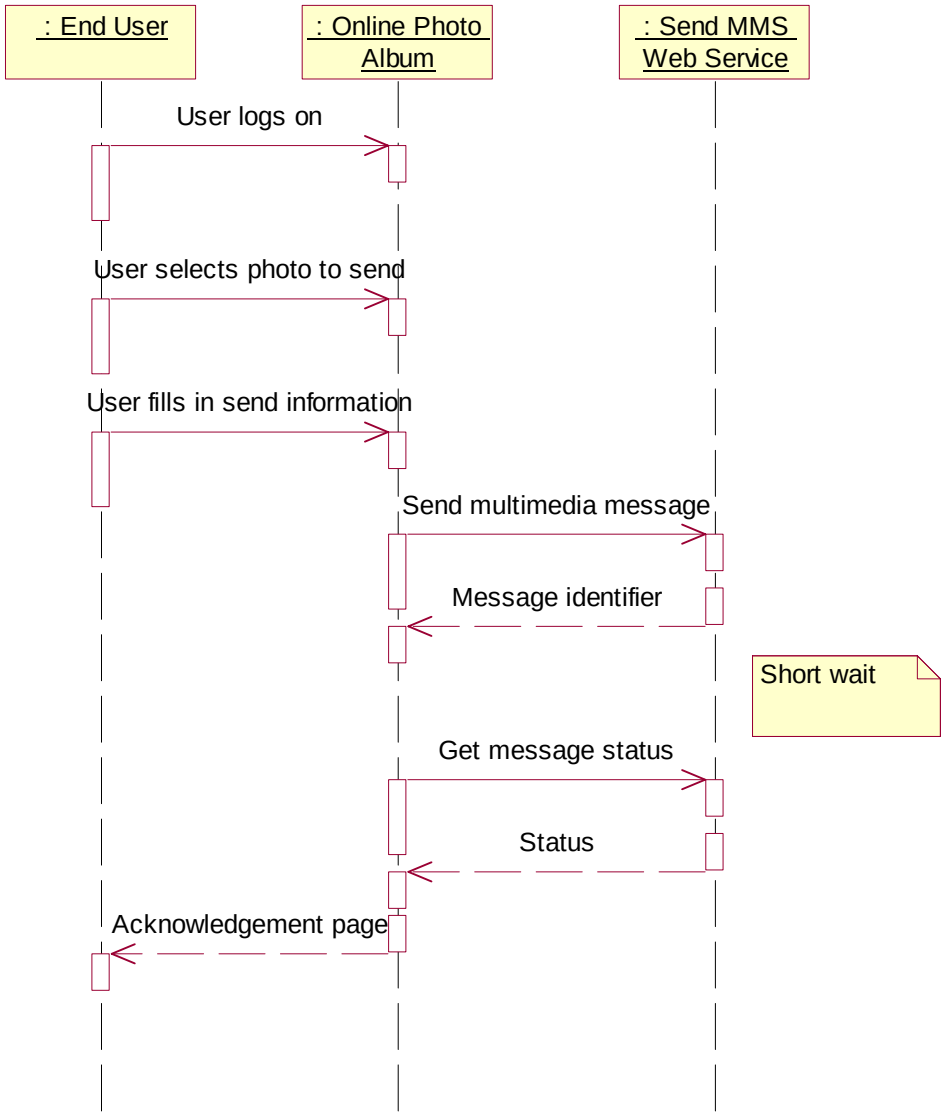


Figure 2

7 XML Schema data type definition

7.1 DeliveryStatus enumeration

List of delivery status values.

Enumeration	Description
Delivered	Successful delivery.
DeliveryUncertain	Delivery status unknown: e.g. because it was handed off to another network.
DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.

7.2 MessagePriority enumeration

List of delivery priority values.

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

7.3 DeliveryInformation structure

Delivery status information.

Element name	Element type	Description
address	xsd:anyURI	Address associated with the delivery status. The address field is coded as a URI.
deliveryStatus	DeliveryStatus	Parameter indicating the delivery status.

7.4 MessageReference structure

Message information.

Element name	Element type	Description
messageIdentifier	xsd:string	OPTIONAL: If present, contains a reference to a message stored in the Parlay X gateway. If the message is pure text, this parameter is not present.
messageService ActivationNumber	xsd:string	Number associated with the invoked Message service, i.e. the destination address used by the terminal to send the message.
senderAddress	xsd:anyURI	Indicates message sender address.
subject	xsd:string	OPTIONAL: If present, indicates the subject of the received message. This parameter will not be used for SMS services.
priority	MessagePriority	The priority of the message: default is Normal.
message	xsd:string	OPTIONAL: If present, then the **messageIdentifier** is not present and this parameter contains the whole message. The type of the message is always pure ASCII text in this case. The message will not be stored in the Parlay X gateway.

7.5 MessageURI structure

Message location information.

Element name	Element type	Description
bodyText	xsd:string	Contains the message body if it is encoded as ASCII text.
fileReferences	xsd:anyURI [0..unbounded]	This is an array of URI references to all the attachments in the Multimedia message. These are URIs to different files, e.g. GIF pictures or pure text files.

8 Web Service interface definition

8.1 Interface: SendMessage

Operations to send messages and check status on sent messages.

8.1.1 Operation: SendMessage

Request to send a Message to a set of destination addresses, returning a **requestIdentifier** to identify the message. The **requestIdentifier** can subsequently be used by the application to poll for the message status, i.e. using **getMessageDeliveryStatus** to see if the message has been delivered or not. The content is sent as an Attachment as specified in SOAP Messages with Attachments [3].

Addresses may include group URIs as defined in the Address List Management specification. If groups are not supported, a PolicyException (POL0006) will be returned to the application.

8.1.1.1 Input message: SendMessageRequest

Part name	Part type	Description
Addresses	xsd:anyURI [0..unbounded]	Destination addresses for the Message.
SenderAddress	xsd:string	Message sender address. This parameter is not allowed for all 3rd party providers. Parlay X server needs to handle this according to a SLA for the specific application and its use can therefore result in a PolicyException. (optional)
Subject	xsd:string	Message subject. If mapped to SMS this parameter will be used as the senderAddress, even if a separate senderAddress is provided. (optional)
Priority	MessagePriority	Priority of the message. If not present, the network will assign a priority based on an operator policy. (optional)
Charging	Common:Charging Information	Charging to apply to this message. (optional)

NOTE: The input message may also contain attachments, with appropriate content as defined by SOAP Messages with Attachments [3].

8.1.1.2 Output message: SendMessageResponse

Part name	Part type	Description
RequestIdentifier	xsd:string	It is a correlation identifier that is used in a **getMessageDeliveryStatus** message invocation, i.e. to poll for the delivery status of all of the sent Messages.

8.1.1.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.
- SVC0004 - No valid addresses.
- SVC0006 - Invalid group.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.
- POL0006 - Groups not allowed.
- POL0007 - Nested groups not allowed.
- POL0008 - Charging not supported.

8.1.2 Operation: GetMessageDeliveryStatus

This is a poll method used by the application to retrieve delivery status for each message sent as a result of a previous **sendMessage** message invocation. The **requestIdentifier** parameter identifies this previous message invocation.

8.1.2.1 Input message: GetMessageDeliveryStatusRequest

Part name	Part type	Description
RequestIdentifier	xsd:string	Identifier related to the delivery status request.

8.1.2.2 Output message: GetMessageDeliveryStatusResponse

Part name	Part type	Description
DeliveryStatus	DeliveryInformation [0..unbounded]	It is an array of status of the messages that were previously sent. Each array element represents a sent message: i.e. its destination address and its delivery status.

8.1.2.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.

8.2 Interface: ReceiveMessage

Operations to retrieve messages that have been received.

8.2.1 Operation: GetReceivedMessages

This method enables the application to poll for new messages associated with a specific **registrationIdentifier**. If the **registrationIdentifier** is not specified, the Parlay X server will return references to all messages sent to the application. The process of binding different **registrationIdentifier** parameters to applications is an off-line process. The Parlay X gateway shall not allow an application to poll for messages using **registrationIdentifier** parameters that are not associated with the application. The priority parameter may be used by the application to retrieve references to higher priority messages, e.g. if Normal is chosen only references to high priority and normal priority messages are returned. If the priority parameter is omitted all message references are returned.

8.2.1.1 Input message: GetReceivedMessagesRequest

Part name	Part type	Description
RegistrationIdentifier	xsd:string	Identifies the off-line provisioning step that enables the application to receive notification of Message reception according to specified criteria.
Priority	MessagePriority	OPTIONAL. The priority of the messages to poll from the Parlay X 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.

8.2.1.2 Output message: GetReceivedMessagesResponse

Part name	Part type	Description
Messages	MessageReference [0..unbounded]	It contains an array of messages received according to the specified filter of **registrationIdentifier** and **priority**.

8.2.1.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.

8.2.2 Operation: GetMessageURIs

This method will read the different parts of the message, create local files in the Parlay Gateway and return URI references to them. The application can then simply read each file or just have them presented as links to the end-user. The URIs to the files will be active for an agreed time.

8.2.2.1 Input message: GetMessageURIsRequest

Part name	Part type	Description
MessageRefIdentifier	xsd:string	The identity of the message to retrieve.

8.2.2.2 Output message: GetMessageURIsResponse

Part name	Part type	Description
Message	MessageURI	It contains the complete message, i.e. the textual part of the message, if such exists, and a list of file references for the message attachments, if any.

8.2.2.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.

8.2.3 Operation: GetMessage

This method will read the whole message. The data is returned as an attachment, as defined in SOAP Messages with Attachments [3], in the return message.

8.2.3.1 Input message: GetMessageRequest

Part name	Part type	Description
MessageRefIdentifier	String	The identity of the message

8.2.3.2 Output message: GetMessageResponse

Part name	Part type	Description
None		

8.2.3.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.

8.3 Interface: MessageNotification

8.3.1 Operation: NotifyMessageReception

This method will have to be implemented by a Web Service on the client application side. The registration of the URI for this application Web Service is done off-line. This means that there is a registration mechanism in the Parlay X Gateway that binds different **registrationIdentifier** parameters to applications and their Web Service URIs.

A client application is notified that a new Message, sent to a specific Service Activation Number, has been received. Using the **registrationIdentifier**, the client application can apply appropriate service logic with specific behaviour.

8.3.1.1 Input message: NotifyMessageReceptionRequest

Part name	Part type	Description
RegistrationIdentifier	xsd:string	A handle connected to the off-line registration of the notifications. This distinguishes registrations that point to the same application Web Service.
Message	MessageReference	This parameter contains all the information associated with the received message.

8.3.1.2 Output message: NotifyMessageReceptionResponse

Part name	Part type	Description
None		

8.3.1.3 Referenced faults

None.

9 Fault definitions

9.1 ServiceException

No new faults are defined by this service.

10 Service policies

Service policies for this service.

Name	Type	Description
GroupSupport	xsd:Boolean	Groups may be included with addresses
NestedGroupSupport	xsd:Boolean	Are nested groups supported in group definitions
ChargingSupported	xsd:Boolean	Charging supported for send message operation

Annex A (normative): WSDL for Multimedia Messaging

The document/literal WSDL representation of this interface specification is compliant to ES 202 391-1 [2] and is contained in text files (contained in archive es_20239105v010101m0.zip) which accompany the present document.

Annex B (informative): Bibliography

ETSI TR 121 905: "Universal Mobile Telecommunications System (UMTS); Vocabulary for 3GPP Specifications (3GPP TR 21.905)".

History

Document history			
V1.1.1	January 2005	Membership Approval Procedure	MV 20050318: 2005-01-18 to 2005-03-18