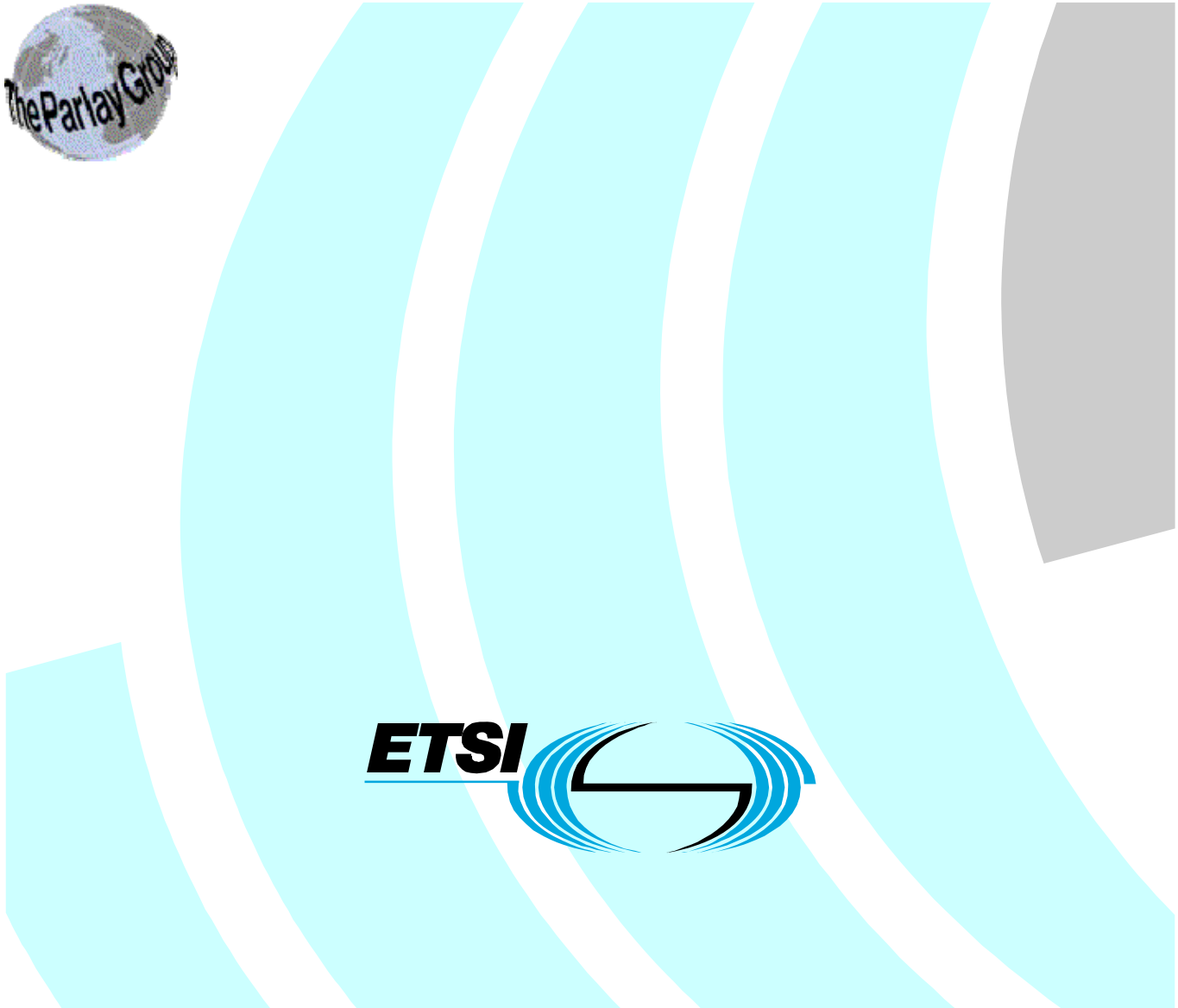
ETSI ES 202 391-4 V1.3.1 (2008-05)

ETSI Standard

Open Service Access (OSA);
Parlay X Web Services;
Part 4: Short Messaging
(Parlay X 2)



Reference
RES/TISPAN-01056-04-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).

The present document is part 4 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 forms part of the Parlay X 2.2 set of specifications.

The present document is equivalent to 3GPP TS 29.199-04 V6.8.0 (Release 6).

1 Scope

The present document is part 4 of the Stage 3 Parlay X 2 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 Short 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

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.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

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

For online referenced documents, information sufficient to identify and locate the source shall be provided. Preferably, the primary source of the referenced document should be cited, in order to ensure traceability. Furthermore, the reference should, as far as possible, remain valid for the expected life of the document. The reference shall include the method of access to the referenced document and the full network address, with the same punctuation and use of upper case and lower case letters.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[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 (Parlay X 2)".

[3] ETSI TS 123 040: "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Technical realization of Short Message Service (SMS) (3GPP TS 23.040)".

[4] IETF RFC 2822: "Internet Message Format".

NOTE: Available at: http://www.ietf.org/rfc/rfc2822.txt

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] and the following apply:

Whitespace: See definition for CFWS as defined in RFC 2822 [].

3.2 Abbreviations

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

SMS	Short Message Service
SMS-C	Short Message Service - Centre

4 Detailed service description

Currently, in order to programmatically receive and send SMS it is necessary to write applications using specific protocols to access SMS functions provided by network elements (e.g. SMS-C). This approach requires a high degree of network expertise. Alternatively it is possible to use the Parlay/OSA approach, invoking standard interfaces (e.g. User Interaction or Messaging Service Interfaces) to gain access to SMS capabilities, but these interfaces are usually perceived to be quite complex by IT application developers. Developers must have advanced telecommunication skills to use OSA interfaces.

In this clause is described a Parlay X 2 Web Service, for sending and receiving SMS messages. The overall scope of this Web Service is to provide to application developers primitives to handle SMS in a simple way. In fact, using the SMS Web Service, application developers can invoke SMS functions without specific Telco knowledge.

ShortMessaging provides operations (see clause 8.1, SendSms API) for sending an SMS message to the network and a polling mechanism for monitoring the delivery status of a sent SMS message. It also provide an asynchronous notification mechanism for delivery status (see clause 8.2.2, SmsNotification API: notifySmsDeliveryReceipt operation).

ShortMessaging also allows an application to receive SMS messages. Both a polling (see clause 8.3, ReceiveSms API) and an asynchronous notification mechanism (see clause 8.2.1, SmsNotification API: notifySmsReception operation and clause 8.4, SmsNotificationManager API) are available.

Figure 1 shows a scenario using the SMS Web Service to send an SMS message from an application. The application invokes a Web Service to retrieve a weather forecast for a subscriber (1) and (2) and a Parlay X 2 Interface (3) to use the SMS Web Service operations (i.e. to send an SMS). After invocation, the SMS Web Service invokes a Parlay API method (4) using the Parlay/OSA SCS (Generic User Interaction) interface. This SCS handles the invocation and sends an UCP operation (5) to an SMS-C. Subsequently the weather forecast is delivered (6) to the subscriber.

In an alternative scenario, the Parlay API interaction involving steps (4) and (5) could be replaced with a direct interaction between the SMS Web Service and the Mobile network.

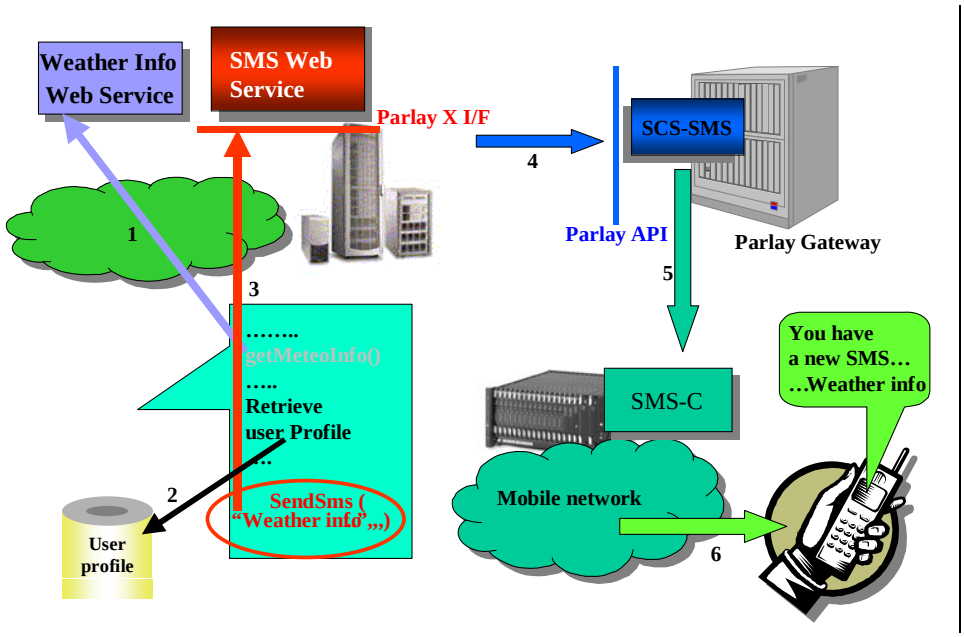


Figure 1: Send SMS Scenario

Figure 2 shows a scenario using the SMS Web Service to deliver a received SMS message to an application. The application receives a Parlay X 2 Web Service invocation for an SMS sent by a subscriber (1) and (2). The SMS message contains the e-mail address of the person the user wishes to call. The application invokes a Parlay X Interface (3) to the Third Party Call Web Service in order to initiate the call (4).

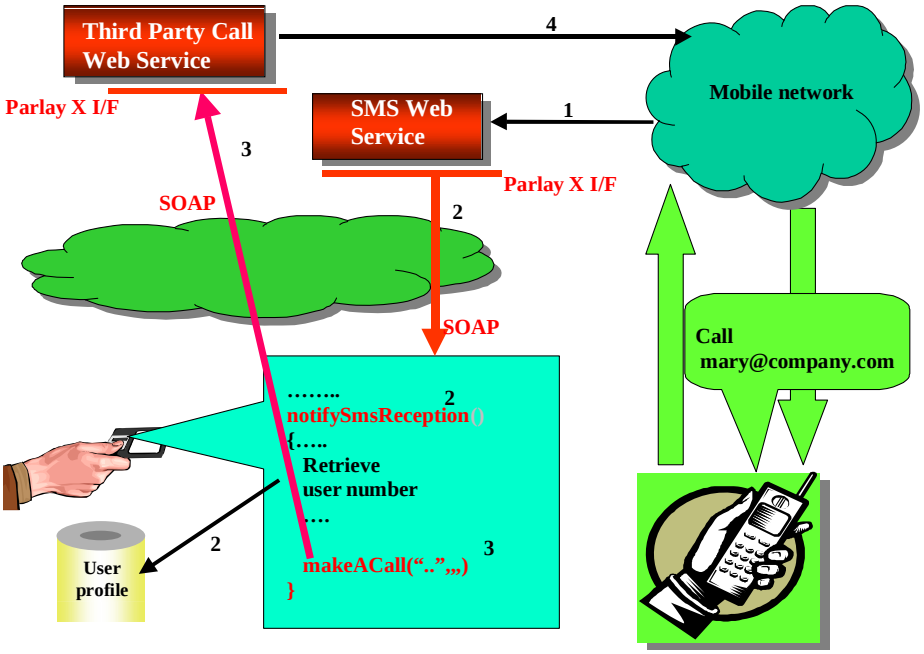


Figure 2: Receive SMS Scenario

5 Namespaces

The SendSms interface uses the namespace:

http://www.csapi.org/wsdl/parlayx/sms/send/v2_3

The ReceiveSms interface uses the namespace:

http://www.csapi.org/wsdl/parlayx/sms/receive/v2_3

The SmsNotification interface uses the namespace:

http://www.csapi.org/wsdl/parlayx/sms/notification/v2_2

The SmsNotificationManager interface uses the namespace:

http://www.csapi.org/wsdl/parlayx/sms/notification_manager/v2_4

The data types are defined in the namespace:

http://www.csapi.org/schema/parlayx/sms/v2_2

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 SMS and report status

Sending SMS message from Web portals is a common capability offered by Service Providers. This sequence diagram shows a portal providing this service.

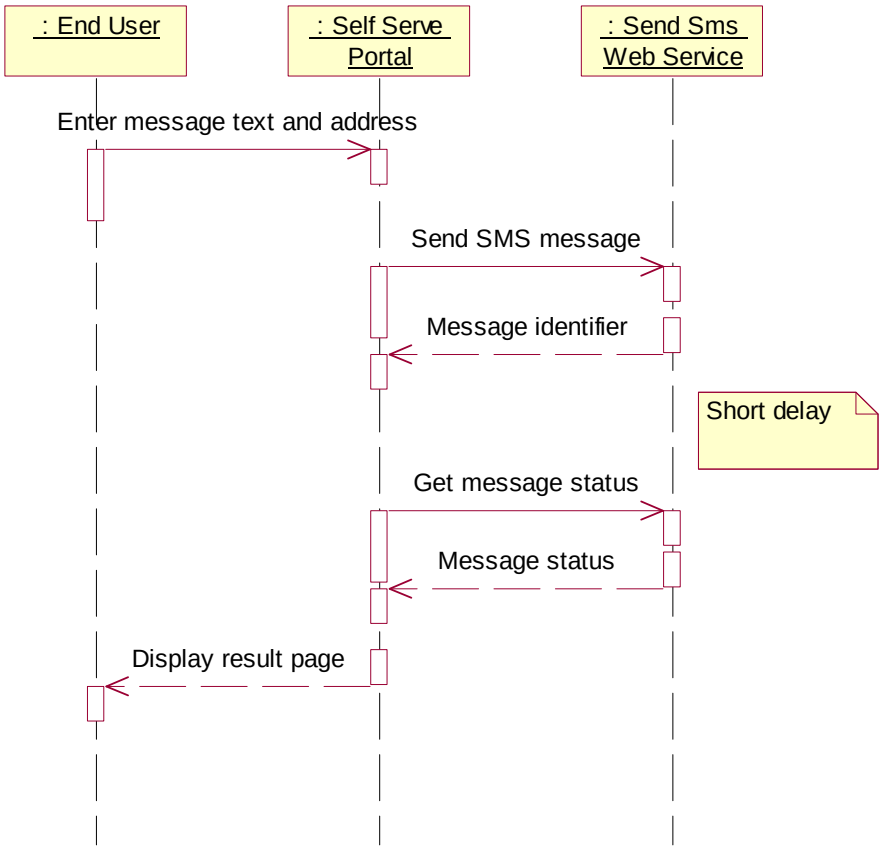


Figure 3

7 XML Schema data type definition

7.1 DeliveryStatus enumeration

List of delivery status values.

Enumeration value	Description
DeliveredToNetwork	Successful delivery to network.
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.
DeliveredToTerminal	Successful delivered to Terminal
DeliveryNotificationNotSupported	Unable to provide delivery receipt notification. The **notifySMSDeliveryReceipt** operation will return "DeliveryNotificationNotSupported" to indicate that delivery receipt for the specified address in a **sendSMS…Request** message is not supported.

7.2 SmsFormat enumeration

List of SMS format values.

Enumeration value	Description
Ems	Enhanced Messaging Service, standardized in TS 123 040 [], which defines a logo/ringtone format
SmartMessaging™	Defines a logo/ringtone format

7.3 DeliveryInformation structure

Delivery status information.

Element name	Element type	Optional	Description
address	xsd:anyURI	No	It indicates the destination address to which the notification is related
deliveryStatus	DeliveryStatus	No	Indicates the delivery result for the destination address

7.4 SmsMessage structure

SMS message information. The **senderAddress** is the address from which the message was actually sent, which may or may not match the **senderName** value provided in the **sendSms** operation.

Element name	Element type	Optional	Description
message	xsd:string	No	Text received in SMS
senderAddress	xsd:anyURI	No	It indicates address sending the SMS
smsServiceActivation Number	xsd:anyURI	No	Number associated with the invoked Message service, i.e. the destination address used to send the message
dateTime	xsd:dateTime	Yes	Time when message was received by operator

8 Web Service interface definition

8.1 Interface: SendSms

This interface defines operations to send various types of Short Messages and to subsequently poll for delivery status. The Short Message types are:

- SMS message, as described in clause 8.1.1.
- SMS logo, as described in clause 8.1.2.
- SMS ringtone, as described in clause 8.1.3.

The send operations for the Short Message types are similar. A description of the common message parts follows:

- **addresses** specifies the destination address or address set for the Short Message. It 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.
- **senderName** is optional and specifies the sender's name: i.e. the string that is displayed on the user's terminal as the originator of the message.
- **charging** specifies the charging information.

- **receiptRequest** is optional and is specified when the application requires to receive notification of the status of the SMS delivery. It is a **SimpleReference** structure that indicates the application endpoint, interface used for notification of delivery receipt and a correlator that uniquely identifies the sending request:
  - If the notification mechanism is not supported by a network, a **ServiceException** (SVC0283) will be returned to the application and the message will not be sent to the addresses specified.
  - The **correlator** provided in the r**eceiptRequest** must be unique for this Web Service and application at the time the notification is initiated, otherwise a **ServiceException** (SVC0005) will be returned to the application.
  - Notification to the application is done by invoking the **notifySmsDeliveryReceipt** operation at the endpoint specified in **receiptRequest**.
- **requestIdentifier** is specified in the response message associated with each send operation. The application can use it to invoke the **getSmsDeliveryStatus** operation to poll for the delivery status.

8.1.1 Operation: sendSms

The application invokes the **sendSms** operation to send an SMS message, specified by the String **message.**

If **message** is longer than the maximum supported length, the message content will be sent as several concatenated short messages.

8.1.1.1 Input message: sendSmsRequest

Part name	Part type	Optional	Description
addresses	xsd:anyURI [1..unbounded]	No	Addresses to which the SMS will be sent
senderName	xsd:string	Yes	If present, it indicates the SMS sender name, i.e. the string that is displayed on the user's terminal as the originator of the message
charging	common:ChargingInformation	Yes	Charge to apply to this message
message	xsd:string	No	Text to be sent in SMS
receiptRequest	common:SimpleReference	Yes	It defines the application endpoint, interfaceName and correlator that will be used to notify the application when the message has been delivered to the terminal or if delivery is impossible

8.1.1.2 Output message: sendSmsResponse

Part name	Part type	Optional	Description
result	xsd:string	No	It identifies a specific SMS delivery request

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.
- SVC0280 - Message too long.
- SVC0283 - Delivery Receipt Notification not supported.

PolicyException from ES 202 391-1 [2]:

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

8.1.2 Operation: sendSmsLogo

The application invokes the **sendSmsLogo** operation to send an SMS logo, specified by the byte array **image**.

8.1.2.1 Input message: sendSmsLogoRequest

Part name	Part type	Optional	Description
addresses	xsd:anyURI [1..unbounded]	No	Addresses to which the SMS logo will be sent
senderName	xsd:string	Yes	SMS sender name, i.e. the string that is displayed on the user's terminal as the originator of the message
charging	common:ChargingInformation	Yes	Charge to apply to this message
image	xsd:base64Binary	No	The image in jpeg, gif or png format. The image will be scaled to the proper format
smsFormat	SmsFormat	No	Conversion to be applied to the message prior to delivery. Possible values are: **"Ems"** or **"SmartMessaging"**
receiptRequest	common:SimpleReference	Yes	It defines the application endpoint, interfaceName and correlator that will be used to notify the application when the message has been delivered to the terminal or if delivery is impossible.

8.1.2.2 Output message: sendSmsLogoResponse

Part name	Part type	Optional	Description
result	xsd:string	No	It identifies a specific SMS delivery request

8.1.2.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.
- SVC0004 - No valid addresses.
- SVC0006 - Invalid group.
- SVC0281 - Unrecognized data format.
- SVC0283 - Delivery Receipt Notification not supported.

PolicyException from ES 202 391-1 [2]:

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

8.1.3 Operation: sendSmsRingtone

The application invokes the **sendSmsRingtone** operation to send an SMS ringtone, specified by the String **ringtone** (in RTX format).

Depending on the length of the ringtone, it may be sent as several concatenated short messages.

NOTE: In the RTX Ringtone Specification, an RTX file is a text file, containing the ringtone name, a control subclause and a subclause containing a comma separated sequence of ring tone commands.

8.1.3.1 Input message: sendSmsRingtoneRequest

Part name	Part type	Optional	Description
addresses	xsd:anyURI [1..unbounded]	No	Addresses to which the SMS logo will be sent
senderName	xsd:string	Yes	SMS sender name, i.e. the string that is displayed on the user's terminal as the originator of the message
charging	common:ChargingInformation	Yes	Charge to apply to this message
ringtone	xsd:string	No	The ringtone in RTX format (see note in clause 8.1.3). (http://www.logomanager.co.uk/help/Edit/RTX.html)
smsFormat	SmsFormat	No	Conversion to be applied to the message prior to delivery. Possible values are: "Ems" or "SmartMessaging"
receiptRequest	common:SimpleReference	Yes	It defines the application endpoint, interfaceName and correlator that will be used to notify the application when the message has been delivered to the terminal or if delivery is impossible

8.1.3.2 Output message: sendSmsRingtoneResponse

Part name	Part type	Optional	Description
result	xsd:string	No	It identifies a specific SMS delivery request

8.1.3.3 Referenced faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.
- SVC0004 - No valid addresses.
- SVC0006 - Invalid group.
- SVC0281 - Unrecognized data format.
- SVC0283 - Delivery Receipt Notification not supported.

PolicyException from ES 202 391-1 [2]:

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

8.1.4 Operation: getSmsDeliveryStatus

The application invokes the **getSmsDeliveryStatus** operation to request the status of a previous SMS delivery request identified by **requestIdentifier**. The information on the status is returned in **deliveryStatus**, which is an array of status related to the request identified by **requestIdentifier**. The status is identified by a couplet indicating a user address and the associated delivery status. This method can be invoked multiple times by the application even if the status has reached a final value. However, after the status has reached a final value, status information will be available only for a limited period of time as defined by a service policy. The following five different SMS delivery status values have been identified:

- **DeliveredToNetwork:** in case of concatenated messages, only when all the SMS-parts have been successfully delivered to the network.
- **DeliveryUncertain:** 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.
- **DeliveredToTerminal:** in case of concatenated messages, only when all the SMS-parts have been successfully delivered to the terminal.

8.1.4.1 Input message: getSmsDeliveryStatusRequest

Part name	Part type	Optional	Description
requestIdentifier	xsd:string	No	It identifies a specific SMS delivery request

8.1.4.2 Output message: getSmsDeliveryStatusResponse

Part name	Part type	Optional	Description
result	DeliveryInformation [0..unbounded]	Yes	It lists the variations on the delivery status of the SMS. Possible values are: • DeliveredToNetwork • DeliveryUncertain • DeliveryImpossible • MessageWaiting • DeliveredToTerminal

8.1.4.3 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.
- POL0010 – Retention time interval expired.

8.2 Interface: SmsNotification

SmsNotification is the application side interface to which notifications about short messages are delivered.

8.2.1 Operation: notifySmsReception

The notification is used to send a short message to the application. The notification will occur only if the SMS fulfils the criteria specified when starting the SMS notification.

The **notifySmsReception** method must be implemented by a Web Service at the ***application side***. It will be invoked by the Parlay X 2 server to notify the application of the reception of an SMS. The notification will occur if and only if the SMS received fulfils the criteria specified in a provisioning step, identified by the **correlator**. The criteria must at least include an **smsServiceActivationNumber**, i.e. the SMS destination address that can be "monitored" by the application. The parameter **senderAddress** contains the address of the sender. The application can apply the appropriate service logic to process the SMS.

8.2.1.1 Input message: notifySmsReceptionRequest

Part name	Part type	Optional	Description
correlator	xsd:string	No	Correlator provided in request to set up this notification
message	SmsMessage	No	Message received

8.2.1.2 Output message: notifySmsReceptionResponse

Part name	Part type	Optional	Description
None			

8.2.1.3 Referenced faults

None.

8.2.2 Operation: notifySmsDeliveryReceipt

The **notifySmsDeliveryReceipt** operation must be implemented by a Web Service at the ***application side*** if it requires notification of SMS delivery receipt. It will be invoked by the Parlay X 2 server to notify the application when a SMS sent by an application has been delivered to the terminal of the recipient or if delivery is impossible. The notification will occur if and only if the status of the sent SMS is **DeliveredToTerminal** or **DeliveryImpossible** and the application has specified interest in notification when sending an SMS message by specifying the optional **receiptRequest** part. The correlator returned corresponds to the identifier specified by the application in the **receiptRequest** of the original **sendSMS[Logo/Ringtone]** request.

When a SMS message is sent to multiple addresses, the server will send a notification for each terminal as and when a SMS message is delivered to the terminal.

The following three different SMS delivery status values will be returned in a n**otifySMSDeliveryReceiptResponse** message:

- **DeliveryImpossible**: unsuccessful delivery; the message could not be delivered before it expired.
- **DeliveredToTerminal**: in case of concatenated messages, only when all the SMS-parts have been successfully delivered to the terminal.
- **DeliveredNotificationNotSupported**: if notification is supported by the network but it does not support delivery receipt for one or more addresses specified in the **sendSMS[Logo/Ringtone]Request** message. The service will send this status for those addresses.

8.2.2.1 Input message: notifySmsDeliveryReceiptRequest

Part name	Part type	Optional	Description
correlator	xsd:string	No	The identifier defining the original "Send" Request. This correlator was provided by the application in the **sendSMS[Logo/Ringtone]Request** message
deliveryStatus	DeliveryInformation	No	It lists the variations on the delivery status of the SMS to a terminal. Possible values are: • DeliveryImpossible • DeliveredToTerminal • DeliveryNotificationNotSupported

8.2.2.2 Output message: notifySmsDeliveryReceiptResponse

Part name	Part type	Optional	Description
None			

8.2.2.3 Referenced faults

None.

8.3 Interface: ReceiveSms

8.3.1 Operation: getReceivedSms

The invocation of **getReceivedSms** retrieves all the SMS messages received that fulfil the criteria identified by **registrationIdentifier**. The method returns only the list of SMS messages received since the previous invocation of the same method, i.e. each time the method is executed the messages returned are removed from the server. Moreover, each SMS message will be automatically removed from the server after a maximum time interval as defined by a service policy.

The received SMS messages are returned in the **getReceivedSmsResponse** message. An SMS message is identified by a structure indicating the sender of the SMS message and the content.

8.3.1.1 Input message: getReceivedSmsRequest

Part name	Part type	Optional	Description
registrationIdentifier	xsd:string	No	Identifies the provisioning step that enables the application to receive notification of SMS reception according to specified criteria

8.3.1.2 Output message: getReceivedSmsResponse

Part name	Part type	Optional	Description
result	SmsMessage [0..unbounded]	Yes	It lists the received SMS since last invocation

8.3.1.3 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.
- POL0010 – Retention time interval expired.

8.4 Interface: SmsNotificationManager

The short message notification manager enables applications to set up and tear down notifications for short messages online.

8.4.1 Operation: startSmsNotification

This operation initiates notifications to the application for a given SMS Service activation number and criteria.

The **smsServiceActivationNumber** is an Address Data item, e.g. a Shortcode, as defined in ES 202 391-1 [2].

The **correlator** provided in the **reference** must be unique for the application Web Service at the time the notification is initiated, otherwise a fault (SVC0005) will be returned to the application.

If specified, criteria will be used to filter messages that are to be delivered to an application. The use of criteria will allow different notification endpoints to receive notifications for the same **smsServiceActivationNumber**. If criteria are not provided, or is an empty string, then all messages for the **smsServiceActivationNumber** will be delivered to the application. If **criteria** values overlap then SVC0008 will be returned to the application and the notification will not be set up. The combination of **smsServiceActivationNumber** and **criteria** must be unique, so that a notification will be delivered to only one notification endpoint. If no match is found, the message will not be delivered to the application.

8.4.1.1 Input message: startSmsNotificationRequest

Part name	Part type	Optional	Description
reference	common:SimpleReference	No	Notification endpoint definition
smsServiceActivationNumber	xsd:anyURI	No	The destination address of the short message
criteria	xsd:string	Yes	The text to match against to determine the application to receive the notification. This text is matched against the first word in the message, defined as the initial characters after discarding any leading Whitespace and ending with a Whitespace or end of message. The matching shall be case-insensitive.

8.4.1.2 Output message: startSmsNotificationResponse

Part Name	Part Type	Optional	Description
None			

8.4.1.3 Referenced Faults

ServiceException from ES 202 391-1 [2]:

- SVC0001 - Service error.
- SVC0002 - Invalid input value.
- SVC0005 - Duplicate correlator.
- SVC0008 - Overlapping Criteria.

PolicyException from ES 202 391-1 [2]:

- POL0001 - Policy error.

8.4.2 Operation: stopSmsNotification

The application may end a short message notification using this operation.

8.4.2.1 Input message: stopSmsNotificationRequest

Part name	Part type	Optional	Description
correlator	xsd:string	No	Correlator of request to end

8.4.2.2 Output message: stopSmsNotificationResponse

Part Name	Part Type	Optional	Description
None			

8.4.2.3 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.

9 Fault definitions

9.1 ServiceException

9.1.1 SVC0280: Message too long

Name	Description
messageId	SVC0280
text	Message too long. Maximum length is %1 characters
variables	%1 Number of characters allowed in a message

9.1.2 SVC0281: Unrecognized data format

Name	Description
messageId	SVC0281
text	Data format not recognized for message part %1
variables	%1 Message part with the unrecognized data

9.1.3 Void

The fault code SVC0282 is reserved and shall not be used.

9.1.4 SVC0283: Delivery Receipt Notification not supported

Name	Description
messageId	SVC0283
text	Delivery Receipt Notification not supported
variables	

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	Is charging supported for send operations
StatusRetentionTime	common:TimeMetric	A time interval that begins after the status of a short message delivery request has reached a final value. During this interval, the delivery status information remains available for retrieval by the application.
MessageRetentionTime	common:TimeMetric	A time interval that begins after the the receipt of a short message. During this interval, the short message remains available for retrieval by the application.

Annex A (normative): WSDL for Short 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_20239104v010301p0.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	March 2005	Publication
V1.2.1	December 2006	Publication
V1.3.1	February 2008	Membership Approval Procedure MV 20080425: 2008-02-26 to 2008-04-25
V1.3.1	May 2008	Publication