



# Mobile Message Application Protocol Specification

Version 1.0-Draft10

DRAFT

**Mobile Message Application Protocol Specification  
v1.0 Draft10 9-Jan-2002**

? 2001-2002 SMS Forum.

**COPYRIGHT**

All rights reserved. This document or any part thereof may not, without the prior written consent of the SMS Forum, be copied, reprinted or reproduced in any material form including, but without prejudice to the foregoing and not by way of exception photocopying, transcribing, transmitting or storing in any medium or translating into any language, in any form or by any means, including but not limited to, electronic, mechanical, xerographic, optical, magnetic, digital or other methodology.

**DISCLAIMER**

WHILST THE GREATEST CARE HAS BEEN TAKEN TO ENSURE THE ACCURACY OF THE INFORMATION AND DATA CONTAINED HEREIN, SMS FORUM DOES NOT WARRANT THE ACCURACY OR SUITABILITY OF SAME FOR ANY SPECIFIC USE. SMS FORUM EXPRESSLY DISCLAIMS ALL AND ANY LIABILITY TO ANY PERSON, WHETHER A PURCHASER OR OTHERWISE, IN RESPECT OF ANY CONSEQUENCES OF ANYTHING DONE OR OMITTED TO BE DONE BY ANY SUCH PERSON IN PARTIAL OR TOTAL RELIANCE UPON THE WHOLE OR ANY PART OF THE CONTENTS OF THIS PUBLICATION OR ANY DERIVATIVE THEREOF.

THE INFORMATION CONTAINED HEREIN IS BELIEVED TO BE ACCURATE AND RELIABLE. HOWEVER, SMS FORUM ACCEPTS NO RESPONSIBILITY FOR IT'S USE BY ANY MEANS OR IN ANY WAY WHATSOEVER. SMS FORUM SHALL NOT BE LIABLE FOR ANY EXPENSES, COSTS OR DAMAGE THAT MAY RESULT FROM THE USE OF THE INFORMATION CONTAINED HOWSOEVER ARISING IN THIS DOCUMENT OR ANY DERIVATIVE THEREOF.

NOTE 1: THE INFORMATION CONTAINED IN THE WITHIN DOCUMENT AND ANY DERIVATIVE THEREOF IS SUBJECT TO CHANGE WITHOUT NOTICE.

NOTE 2: THE CORPORATE NAME OF SMS FORUM IS NORTHGROVE LIMITED, COMPANY NUMBER 309113, REGISTERED OFFICE GARDNER HOUSE, WILTON PLACE, DUBLIN 2.

# Table Of Contents

1	Introduction.....	8
1.1	Scope Of This Document .....	10
1.2	Overview.....	10
1.2.1	Supported Cellular Technologies.....	10
1.3	Examples .....	10
1.3.1	Text Messaging over HTTP .....	10
1.3.2	MultiMedia Messaging over HTTP .....	11
1.3.3	Mobile Messaging using SOAP and HTTP .....	13
1.4	Glossary .....	16
1.5	References .....	17
2	Design Principles .....	19
2.1	Summary of Approach.....	19
2.1.1	General Principles.....	19
2.1.2	Design Decisions .....	19
2.1.2.1	Modularisation .....	19
2.1.2.2	Language and character-set selection.....	20
2.1.2.3	Operational modes .....	20
2.1.2.4	Network-specific features.....	21
2.1.2.5	Dynamic feature selection.....	21
2.1.2.6	Static feature selection (profiles) .....	21
2.1.3	XML Implementation choices .....	22
2.1.4	XML Naming Guidelines.....	22
3	Procedures .....	23
3.1	Session Management.....	23
3.1.1	Overview .....	23
3.1.1.1	Identification .....	23
3.1.1.2	Authentication.....	23
3.1.1.3	Capability presentation .....	23
3.1.2	Client Session.....	24
3.1.2.1	Example .....	24
3.1.3	Peer-to-peer Session .....	25
3.1.4	Batch processing .....	26
3.2	Message Submission .....	27
3.2.1	Submission to a single destination .....	27
3.2.2	Submission to multiple destinations.....	28
3.2.3	Submission using a distribution list.....	28
3.2.3.1	Example .....	29
3.3	Message Delivery.....	29
3.3.1	Examples .....	30
3.4	Message administration .....	31
3.4.1	Querying Message status.....	31
3.4.2	Modifying a message after submission.....	31
3.4.3	Cancelling a message after submission.....	32
4	XML Data Elements .....	32
4.1	XML Schema document structure.....	32
4.1.1	Specification documents .....	32
4.1.2	Profile Documents.....	33
4.2	Common Definitions.....	33
4.2.1	ApplicationContext .....	33
4.2.2	Response elements .....	34
4.2.2.1	Example .....	34
4.2.3	Billing.....	35
4.2.3.1	Example .....	35
4.3	Session Management.....	35
4.3.1	BindRequest .....	35

4.3.1.1	Example .....	36
4.3.2	BindResponse .....	37
4.3.3	BindBack Request .....	37
4.3.3.1	Example .....	37
4.4	Message Transfer .....	38
4.4.1	Submit.....	38
4.4.2	Deliver .....	38
4.4.3	Expanded Submission Response .....	38
4.4.3.1	Example .....	38
4.4.4	Delivery Failure Response .....	39
4.5	Message Administration .....	39
4.5.1	QueryRequest and QueryResponse .....	39
4.5.1.1	Example .....	40
4.5.2	CancelRequest .....	40
4.5.2.1	Example .....	40
4.5.2.1.1	Cancel by service type .....	40
4.5.2.1.2	Cancel by Reference .....	41
4.5.3	ReplaceRequest .....	41
4.5.3.1	Example .....	41
4.6	Mobile Message Module .....	42
4.6.1	Header.....	42
4.6.1.1	DeliveryOptions .....	42
4.6.1.2	Example .....	43
4.6.2	Body .....	43
4.6.2.1	Examples .....	44
4.6.3	Delivery Info.....	44
4.6.3.1	Example .....	45
4.7	Message Extensions Module.....	45
4.7.1	NetworkHandling Element.....	45
4.7.2	ApplicationOptions element.....	46
4.7.2.1	NumberOfVoiceMails.....	46
4.7.2.2	Callback .....	47
4.7.2.3	ITSSession.....	47
4.7.2.4	ITSReply .....	47
4.7.2.5	Example .....	47
4.7.3	DeviceControl element.....	48
4.7.3.1	GSMOptions.....	48
4.7.3.2	DisplayTime.....	49
4.7.3.3	MSValidity .....	49
4.7.3.4	MessageWaitInd.....	49
4.7.3.5	ReceiveAlert.....	49
4.7.3.6	Example .....	49
4.8	Address Module .....	49
4.8.1	Number.....	50
4.8.1.1	Examples .....	50
4.8.2	Application.....	50
4.8.3	IP .....	50
4.8.4	EMail .....	51
4.8.5	Simple Addressing Examples.....	51
4.8.6	Extended Addressing .....	51
4.8.6.1	Bearer .....	51
4.8.6.2	SubAddress.....	52
4.8.6.3	Examples .....	52
5	Schema Specification .....	53
5.1	Profiles.....	53
5.1.1	BasicPush.xsd .....	53
5.1.2	MMAP.xsd .....	53
5.2	Data definition modules.....	53

5.2.1	Common.xsd.....	53
5.2.2	Session.xsd .....	56
5.2.3	MessageTransfer.xsd.....	58
5.2.4	MessageAdmin.xsd.....	59
5.2.5	Addressing.xsd .....	61
5.2.6	MessageExtensions.xsd.....	65

## List Of Tables

Table 1-1 Glossary	16
Table 1-2 References	18

## List Of Figures

Figure 1-1 MMAP Gateway Configuration .....	8
Figure 1-2 MMAP native-mode SMSC access.....	9
Figure 3: Client Session.....	24
Figure 4: Asynchronous Session.....	26
Figure 5: Batch session.....	27
Figure 6: Submission to a single destination.....	27
Figure 7: submission to multiple destinations .....	28
Figure 8: Distribution List Submission .....	29
Figure 9: Message Delivery.....	30
Figure 10: Query Request.....	31

# 1 Introduction

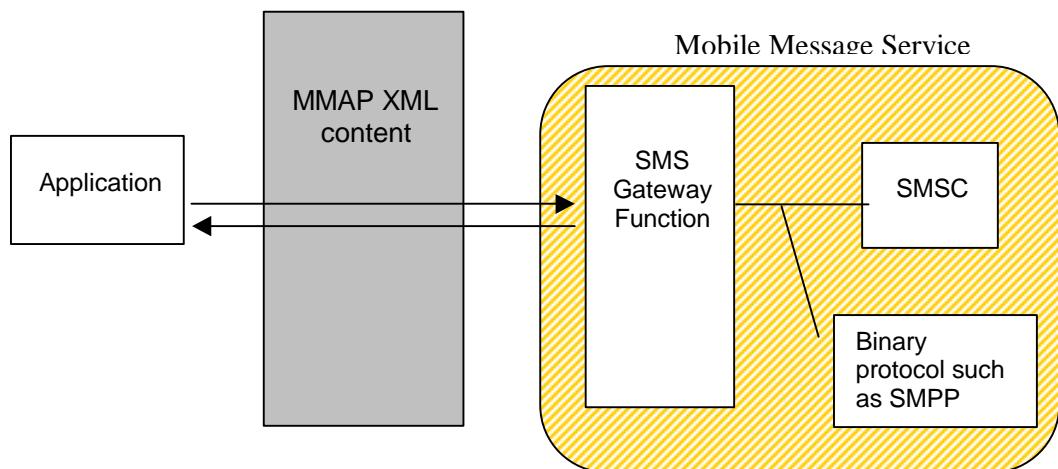
The SMS Forum, a non-profit organisation dedicated to the promotion of SMS within the wireless industry, manages the Mobile Message Application Protocol (MMAP). The specification and related documentation is available from the SMS Forum Website <http://www.smsforum.net>

The Hypertext Transport Protocol (HTTP) is an increasingly popular transport mechanism for sending messages from web applications to mobile phones. Many service providers implement HTTP interfaces. These interfaces are, however, implementation-specific, may offer only a limited set of functions and frequently do not have a full and formal specification.

A number of existing protocols support the transfer of messages between a Short Message Center and External Short Message Entities (ESMEs). These are normally specified as binary protocols accessed over TCP/IP or X.25. It is proposed to define a new text protocol to support application access to message centers and/or messaging gateway nodes over HTTP and other Web protocols. It is further proposed to specify this new protocol in XML. This will allow messages to be specified in a structured format and will also facilitate automatic translation to other messaging formats.

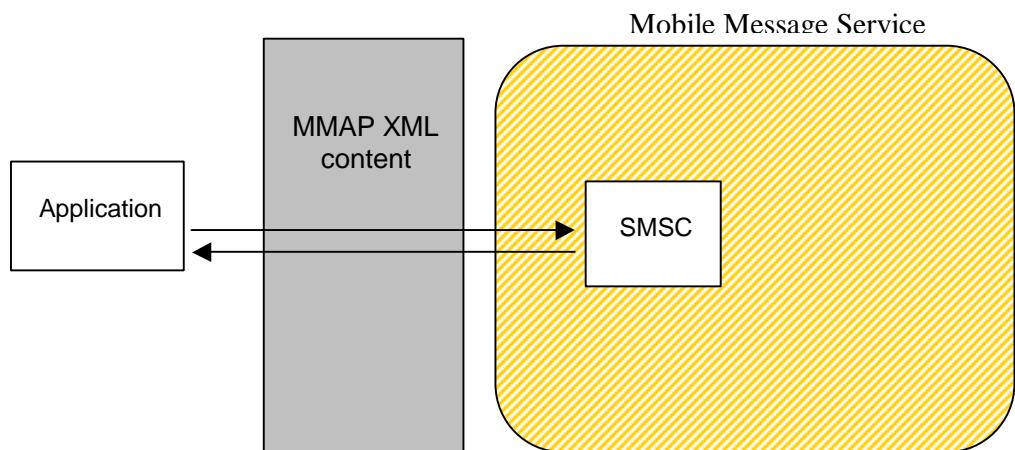
This new Mobile Message Application Protocol (MMAP) is not specified as a one-to-one mapping from any existing binary protocol onto XML elements. It is instead an independent protocol used by applications to deliver messages to a mobile message service. To ensure completeness, this protocol will be validated to ensure that it is functionally equivalent to the SMS forum's binary access protocol; SMPP version 3.4

The following diagram illustrates the context of MMAP in a mobile network using a gateway:



**Figure 1-1 MMAP Gateway Configuration**

The following diagram illustrates the context of MMAP where an SMSC contains a native-mode MMAP access function:



**Figure 1-2 MMAP native-mode SMSC access**

## 1.1 Scope Of This Document

This document defines Version 1.0 of the MMAP protocol. It is intended for designers and implementers of an text protocol interface between Message Service and an external Application.

## 1.2 Overview

The following sub-sections describe briefly the basic concepts and characteristics of MMAP. Many of these characteristics will be discussed in greater detail throughout the document.

### 1.2.1 Supported Cellular Technologies

MMAP is designed to support messaging functionality for any cellular technology and has specific applications and features for technologies such as:

- ?? GSM
- ?? GPRS
- ?? CDMA
- ?? CDMA2000
- ?? UWC-136
- ?? IDEN

## 1.3 Examples

### 1.3.1 Text Messaging over HTTP

The following example shows how MMAP can be used to submit a message. The example uses MMAP in “immediate mode” with HTTP as a bearer. It should be noted that MMAP has other modes of operation and is bearer-independent. HTTP has simply been chosen as a well-known bearer protocol.

```
POST /serv1et/MMAPhandler HTTP/1.1
Host: www.smsforum.net
encoding=text/XML

<?xml version="1.0" encoding="UTF-8"?>
<operation xml:ns="http://www.smsforum.net/mmap"
           xml:ns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.smsforum.net/mmap
basicPush.xsd">

    <ApplicationContext service="test"/>
    <SubmitRequest>
        <MobileMessage>
            <Header>
                <Destination>
                    <Number>+4478912345</Number>
                </Destination>
            </Header>
            <Body>
                <Text>hello world</Text>
            </Body>
        </MobileMessage>
    </SubmitRequest>
</operation>
```

### 1.3.2 MultiMedia Messaging over HTTP

The following example shows how MMAP can be used to submit a multimedia message. The example uses MMAP in “client session mode” with HTTP as a bearer. An ExternalContent element is used with a link to the second MiME multipart that contains the SMIL multipart message

```

POST /serv1 et/MMAPSessionHandler HTTP/1.1
Host: www.smsforum.net
Content-Type: Multi part/Related; boundary=MME_boundary;
type=text/xml;
start=<adsadsad67634kl fl cvreh>
Content-Length: XXXX

--MME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <adsadsad67634kl fl cvreh>

<?xml version="1.0" encoding="UTF-8"?>
<Operation xml ns="http://www.smsforum.net/mmap"
            xml ns: xsi =http://www.w3.org/2001/XMLSchema-instance
            xsi : schemaLocation="http://www.smsforum.net/mmap
mmap.xsd">

    <ApplicationContext service="test2" sessionId="123adf
sourceOperationReference="1"/>
    <SubmitRequest>
        <MobileMessage>
            <Header>
                <Destination>
                    <Number>+4478912345</Number>
                </Destination>
            </Header>
            <Body>
                <ExternalContent link="cid:<mymessage>">
            </Body>
        </MobileMessage>
    </SubmitRequest>
</Operation>

--MME_boundary
Content-Type: application/vnd.wap.multipart.related;
type=application/smil; start=<AA>;
boundary="2882.23513.0=_VaLuE"
Content-ID: <mymessage>

--2882.23513.0=_VaLuE
Content-Type: application/smil
Content-ID: <AA>
Content-Transfer-Encoding: 7bit

<smil>
    <head>
        <layout>
            <region id="imageregion" />
            <region id="textregion1" />
            <region id="textregion2" />
        </layout>
    </head>
    <body>
        <par>
            
            <text src="cid:A1" end="6s" region="textregion1" />
            <text src="cid:A3" begin="6s" region="textregion2" />
        </par>
    </body>

```

```
</smil>
--2882.23513.0=_VaLuE
Content-Type: text/plain
Content-ID: <A1>
Content-Transfer-Encoding: 7bit

MMS text for our example
--2882.23513.0=_VaLuE
Content-Type: image/x-wap.wbmp
Content-ID: <A2>
Content-Transfer-Encoding: base64

FFDEWS///////////////4//////////H////////6f///P////5////
8f///n8D
/+//2eH5/z///ZgAR9H///9gEMzwf///2B/zkx///8P/uPH///+t6fX
8f///5v+f/
D///nwcf4P///+wV54D///+Jg/w///YCJ/D///gIF8P///vBgXw/
///+/+A/D/
///782D8P///3P/w///+ff/D///797/sP///v4f+g///f//2D//
8n/wYf//
///absde///Hv8P///57B/x///P//w///6X38H///nfgAf///
/+G9/h///
/9mH+H///xofwP///4EeA///5c4D///x9wP///n4A///
/P8D///
/9PwH///3DAf///fgB///+/wH///7/gf///+AP///
8P4ACH///+
GPgAA9///+G8GAA7///3f6AIAB///++vn+///77g/7/+3///nwv/v/7///+gP
fy//jf/59+
97/9///4D6ff+8f///QHx//f///9/w/9wA///4A///////////
////
--2882.23513.0=_VaLuE
Content-Type: text/plain
Content-ID: <A3>
Content-Transfer-Encoding: 7bit

This is another piece of normal text included
--2882.23513.0=_VaLuE-
-- MIME_boundary
```

### 1.3.3 Mobile Messaging using SOAP and HTTP

The following example shows how MMAP can be used to bind to a message service over SOAP using HTTP as a bearer.

```
POST servl et/MMAPSOAPhandl er HTTP/1. 1
Host: www. smsforum. net
Content-Type: Muli part/Rel ated; boundary=MM E_boundary;
type=text/xml ;
        start=<tesattatat>
Content-Length: XXXX
SOAPAction: http://www. smsforum. net/protocols/MMAP/SOAP. xsd
Content-Description: MMAP SOAP example.

-- MM E_boundary
Content-Type: text/xml ; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <tesattatat>

<?xml versi on="1. 0" encoding="UTF-8"?>
<soap: Envel ope xml ns: soap="http://www. w3. org/2001/12/soap-
envel ope" xml ns: xsi="http://www. w3. org/2001/XMLSchema- instance"
xsi: schemaLocati on="http://www. w3. org/2001/12/soap-envel ope
C:\Devel opment\investigati ons\xerces\xerces\SOAP. xsd">
    <soap: Header>
        <MMAP: MMAPHeader mustUnderstand="1"
        xml ns: MMAP="http://www. smsforum. net/mmmap"
        xs: schemaLocati on="http://www. smsforum. net/mmmap
        C:\Devel opment\investigati ons\xerces\xerces\SOAP. xsd">
            <MMAP: Appl i cati onContext service="testSOAP"/>
        </MMAP: MMAPHeader>
    </soap: Header>
    <soap: Body>
        <MMAP: MMAPBody mustUnderstand="1"
        xml ns: MMAP="http://www. smsforum. net/mmmap"
        xs: schemaLocati on="http://www. smsforum. net/mmmap
        C:\Devel opment\investigati ons\xerces\xerces\SOAP. xsd">
            <MMAP: Bi ndRequest>
                <MMAP: Sessi onControl >
                    <MMAP: FeatureSet/>
                </MMAP: Sessi onControl >
            </MMAP: Bi ndRequest>
        </MMAP: MMAPBody>
    </soap: Body>
</soap: Envel ope>
```



## 1.4 Glossary

Term	Definition
ACK	Acknowledgement
API	Application Programming Interface
CDMA	Code Division Multiple Access
CDR	Call Detail Record
ETSI	European Telecommunications Standards Institute
GSM	Global System for Mobile Communications
IDEN	Integrated Dispatch Enhanced Network
MC	Message Centre - A generic term used to describe various types of SMS Gateways.
MSB	Most Significant Byte
MSC	Mobile Switching Centre
MS	Mobile Station
MWI	Message Waiting Indication
NACK	Negative Acknowledgement
NSAP	Network Service Access Point
PDU	Protocol Data Unit
SME	Short Message Entity
SMSC	Short Message Service Centre
SMPP	Short Message Peer to Peer Protocol
TETRA	Terrestrial Trunked Radio
TDMA	Time Division Multiple Access
UDHI	User Data Header Indicator
UMTS	Universal Mobile Telecommunications Systems
URL	Uniform Resource Locator
VMA	VoiceMail Alert
TIA	Telecommunications Industry Association
WAP	Wireless Application Protocol ( <a href="http://www.wapforum.org">http://www.wapforum.org</a> )
WCMP	Wireless Control Message Protocol
WDP	Wireless Datagram Protocol

Table 1-1 Glossary

## 1.5 References

Ref.	Document Title	Document Number	Version Number
[GSM 03.40]	Technical Realisation of the Short Message Service Point to Point	GSM 03.40 <a href="http://www.etsi.fr">http://www.etsi.fr</a>	v5.7.1
[GSM 03.38]	“Digital Cellular telecommunications system (Phase 2+); Alphabets and language specific information”.	[GSM 03.38] <a href="http://www.etsi.fr">http://www.etsi.fr</a>	v5.5.1 Sept. '97
[GSM MAP 09.02]	GSM Mobile Application Part	[GSM MAP 09.02] <a href="http://www.etsi.fr">http://www.etsi.fr</a>	v5.11.0
[IS637]	Short Message Service for Spread Spectrum Systems	TIA/EIA/IS-637-A	Rev A
[TSAR]	Teleservice Segmentation and Reassembly (TSAR)	TIA/EIA-136-620	Rev 0
[CMT-136]	Short Message Service - Cellular Messaging Teleservice	TIA/EIA-136-710-A	Rev A
[GUTS]	General UDP Transport Service (GUTS)	TIA/EIA-136-750	Rev 0
[WAPARC H]	Wireless Application Protocol Architecture Specification	WAP Forum <a href="http://www.wapforum.org">http://www.wapforum.org</a>	Version 30-Apr.-1998
[WCMP]	Wireless Control Message Protocol Specification	WAP Forum <a href="http://www.wapforum.org">http://www.wapforum.org</a>	Version 12-June-1998
[WDP]	Wireless Datagram Protocol Specification	WAP Forum <a href="http://www.wapforum.org">http://www.wapforum.org</a>	Version 10-Feb.-1999
[ITUT X.213]	Open Systems Interconnection - Network Service Definition	[ITUT X.213]	11/95
[KOR ITS]	PCS operators common standards for handset-SMS functionalities	PCS standardization committee PCS-SMS-97-05-28	1.06 Rev 99-04-30

Ref.	Document Title	Document Number	Version Number
XML	Extensible Markup Language	W3C ( <a href="http://www.w3c.org/XML">http://www.w3c.org/XML</a> )	1.0

**Table 1-2 References**

## 2 Design Principles

### 2.1 Summary of Approach

#### 2.1.1 General Principles

The following design principles have been established:

**Ease of Use:** Users with little or no knowledge of SMS or telecom protocols should be able to send and receive messages using a simple set of primitives.

**Modularity:** The protocol should be defined as a set of modules, to support interworking with other systems and applications.

**Readability:** Wherever possible attributes and elements should have human readable values rather than "magic" numbers;

**Rich functionality:** All of the functions of a binary access protocol (such as SMPP V3.4) shall be available to advanced users.

**Transport-independence:** it should be possible to provide alternative transport protocols to HTTP, for instance file transfer or SMTP.

#### 2.1.2 Design Decisions

##### 2.1.2.1 Modularisation

The MMAPI specification has been split into modules. The following modules have been defined:

1. **Addressing:** This module defines a set of generic addressing formats for applications. It integrates a number of optional and mandatory addressing parameters as components of a single SMAddress Element.
2. **Mobile Message:** This module provides a generic definition of a mobile message and a delivery information element. Together with the addressing module it provides a protocol-independent definition of a mobile message that can be used as part of any XML-based Mobile Message protocol. (e.g. for onward forwarding of messages within an enterprise).
3. **Basic Operations:** This module defines a basic set of operations for mobile message applications. These provide:
  - a. Session Management
  - b. Message submission
  - c. Message delivery
4. **Message Administration:** This module defines the following MMAPI operations:
  - ?? Message status query
  - ?? Cancel message in transit

?? Replace message contents in transit

5. **Message Extensions:** This module extends the message header and body with extra information for mobile messaging. This includes:

?? Network handling options

?? Application information

?? Device control options

?? User Data Header information

6. **Batch:** This module defines batch request and response operations.

### 2.1.2.2 Language and character-set selection

This is handled as follows:

- a. If the user specifies the message body as containing text, only the language may be specified and the text is standard XML Unicode as defined in The Unicode Standard, Version 2.0. The message service will handle any character-set conversion from Unicode to other network or device-specific encodings (e.g. default GSM alphabet).
- b. If the user specifies the body as BinaryData, the user can specify the encoding of the data explicitly. The user supplies in a hex string the actual data to be sent to the network.
- c. The standard xml:lang attribute shall be used to specify language. The allowable values for this attribute are defined in IETF RFC1766.

### 2.1.2.3 Operational modes

It has been decided to support the following modes of operation for applications:

#### **Immediate mode:**

In this mode the application does not maintain a session with the mobile message service. Each operation contains an application context that specifies access control information and a single request. The message service sends a response only when it has performed the requested operation. The response contains as application information the associated MMAP response element. This mode is used only for message submission.

#### **Client session mode:**

In this mode the application establishes a session with the message service through a bind request. Each operation contains an application context that identifies the session and a single request. The message service sends a response only when it has performed the requested operation. The response contains as application information the associated MMAP response element. The message service can also establish a client session with the application, in order to perform message delivery.

#### **Peer-to-peer session mode**

In this mode, the application and the message service establish a bi-directional session. Once this session is established the message service responds to requests immediately on reception. This response does not contain any application-level information and only indicates reception of the request by the message service. Once the operation has been completed by the gateway, the MMAP response is sent to the client in a separate transport request. The message service can also generate operations asynchronously on the same session, e.g. for message delivery.

#### **Batch mode**

In this mode, the message service receives a set of MMAP requests to process. The MMAP data consists of an application context element followed by a set of requests. The message service processes all of the requests and then generates a new batch element containing the respective response elements. The message service then sends the complete batch response back to the client. The message service can use the same mechanism to perform delivery operations to the application.

Batch mode is normally used when an interactive session is not required or would be unsuitable due to timeout problems. For example, if an application wishes to send an SMS to a distribution list containing many thousands of names, a batch request is probably the most appropriate mechanism.

#### **2.1.2.4 Network-specific features**

asas

#### **2.1.2.5 Dynamic feature selection**

- a. Attribute Entities are defined that identify elements using bearer-specific network features and elements that require different grades of application support.
- b. At bind time, an application indicates the feature support it requires. If the message service does not support these features or does not authorize them for the particular application, the bind is rejected
- c. If a request is made using an element that the session does not support, the message service will either reject the request or remove the element before processing the request.

#### **2.1.2.6 Static feature selection (profiles)**

Selecting features dynamically is extremely flexible but exposes clients to the risk of interoperability problems. A profile is a predefined package of certain features. If a message service implements a profile, any client that conforms to that profile can interoperate with it.

Profiles are defined as separate documents using the elements defined in this specification. It is expected that profiles will additionally define the bearer protocol(s) used, so that a complete definition of how to access a message service is available to the user.

Profiles will be available for the following applications:

- ?? Basic push application
- ?? Interactive client application

?? Interactive peer application

?? Batch application

### 2.1.3 XML Implementation choices

Use of XML requires some overall design decisions to be made. In particular XML offers three different possibilities for handling information within an XML element. Information can be specified as;

1. Body text in the element
2. As another element (i.e. a child of the original element)
3. As an attribute of the element

In this specification the following approach is proposed:

- ?? Where parameters may be extended in the future or can be decomposed into simpler elements, they will be specified as child elements.
- ?? Where parameters can be specified as one of a bounded range of values, these values will be specified as a choice of attribute values.
- ?? Where a parameter is an integer or is a fixed-size binary value of less than 4 octets, it will be specified as an attribute of an element.
- ?? Where a parameter is accessed in more than one place it will normally be defined as an element.
- ?? Where a parameter contains human-readable text or is a larger binary value (> 4 octets) it will normally be specified as the body text of an element.

*Note: the word "normally" in the above list indicates that the recommended approach is not mandatory.*

### 2.1.4 XML Naming Guidelines

The following guidelines are followed in this specification:

- ?? XML element names shall use Upper Camel Case convention, where words are concatenated to form an element name with the first letter of each word in upper case (e.g. ElementName). No hyphens or underscores shall be permitted in the name.
- ?? XML attribute names shall use Lower Camel Case convention, where words are concatenated to form an attribute name. The first word in the element shall be in lower case and all the other words have their first letter in upper case (e.g. attributeName). No hyphens or underscored shall be permitted in the name.
- ?? The only exception to these case conventions is where an acronym is used instead of an English word (e.g. GSM). In this case all of the letters of the acronym shall be in upper case even where the acronym forms the first or only word of an attribute name.

?? All XML schema datatype names shall use the suffix "Type" to distinguish them from elements and attributes.

The XML definitions are specified for a target namespace of <http://www.smsforum.net/mmap>.

## 3 Procedures

### 3.1 Session Management

#### 3.1.1 Overview

A session is established between an application and a message service through a bind procedure. Binding provides a mechanism for identification and authentication of the communicating entities and also provides facilities for identifying the capabilities of each entity and the type of session required. The session type can be one of "Client" or "Peer".

No session is needed where a request is made in immediate mode. Instead, the application includes the required access control information in an application context element with each request.

If requests are made in batch mode, a single application context element provides the access control information. Since all of the information is supplied in a single XML element, there is no need to establish a session between the two entities.

##### 3.1.1.1 Identification

The Bind procedure provides the message service with

1. The name of the application binding to it
2. Account information for any billing associated with the session
3. Origination addressing information
4. Destination addressing information for routing incoming messages to the application.

The bind procedure may also identify application addresses for outgoing binds from the message service.

##### 3.1.1.2 Authentication

In the initial release, the bind procedure provides simple password authentication. If security is an issue, the underlying transport should provide encryption facilities.

It is envisioned that in a future release, more sophisticated certificate-based authentication could be introduced. This could be an independent mechanism or, preferably, could be bound to an underlying security framework.

##### 3.1.1.3 Capability presentation

As part of the bind procedure, the entities exchange capability information. These define

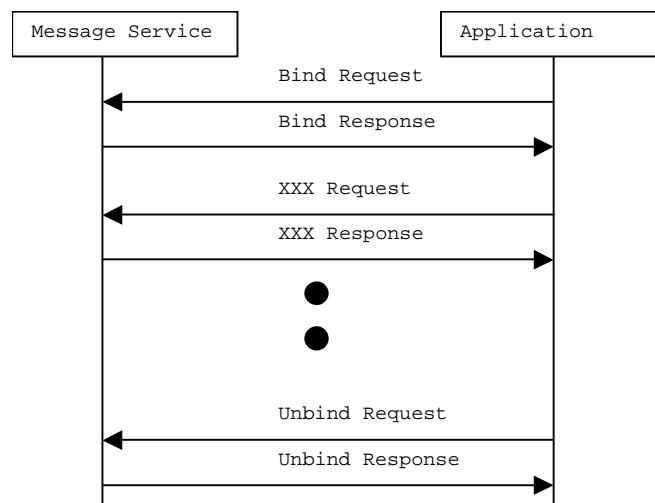
1. The capabilities of the originating entity
2. The facilities required by the originating entity.

This is simply a capability presentation procedure and not a complete capability negotiation process. The action to be taken when a requested facility is not available is up to the communicating entities themselves.

### 3.1.2 Client Session

In a client session, the response to each request is only generated when the request has been performed completely. Either the message service or the application initiates a client session. When asynchronous message submission and delivery is required, two independent client sessions must be established. The SM application initiates a session to perform asynchronous message submission. The message service initiates a second session to perform asynchronous message delivery.

The following diagram shows an application initiating and using a client session.



**Figure 3: Client Session**

#### 3.1.2.1 Example

This XML example shows a simple bind request:

```

<?xml version="1.0" encoding="UTF-8"?>
<nmap:Operation xml:ns:nmap="http://www.smsforum.net/nmap"
  xml:ns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/nmap
  basic_operations.xsd">
  <nmap:ApplicationContext service="testService">
    <nmap:AccessControl>

    <nmap:ApplicationIdentity>testApp</nmap:ApplicationIdentity>
    </nmap:AccessControl>
  </nmap:ApplicationContext>
  <nmap:BindRequest>
    <nmap:SessionControl>
      <nmap:FeatureSet
        featuresRequired="MessageSubmission"
        featuresSupported="MessageSubmission
MessageDelivery"/>
    </nmap:SessionControl>
  </nmap:BindRequest>
</nmap:Operation>

```

The response is encoded as follows:

```

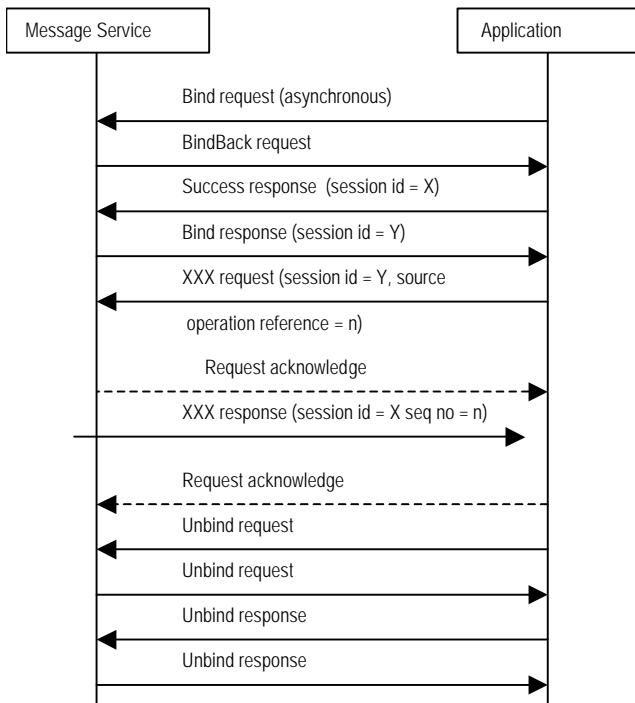
<?xml version="1.0" encoding="UTF-8"?>
<nmap:Operation xml:ns:nmap="http://www.smsforum.net/nmap"
  xml:ns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/nmap
  basic_operations.xsd">
  <nmap:ApplicationContext service="testService"
  sessionID="GGAFC">
    <nmap:SuccessResponse xsi:type="nmap:BindResponseType">
      <nmap:FeatureSet
        featuresSupported="MessageSubmission
MessageDelivery ExtendedAddressing MessageAdministration "
        maxBytesPerMessage="140"/>
    </nmap:SuccessResponse>
</nmap:Operation>

```

### 3.1.3 Peer-to-peer Session

A peer-to-peer session allows an application and message service to exchange information as peers in a single session. Each entity can generate a request asynchronously. The reception of the request is acknowledged immediately and the response is returned asynchronously to the requesting entity. This mode of operation does not map transparently onto a pure client-server protocol such as HTTP, where it is necessary to maintain the bind over two independent client-server sessions. In one of these sessions, the message service acts as a server and the application acts as a client. In the other, the message service acts as a client and the application acts as a server. In order to support this mode of operation, the initial bind may contain additional addressing and security information to allow the other entity to generate a back bind that can be logically linked to the first session. This then allows the two entities to communicate in a peer-to-peer fashion across two independent client-server links.

The following diagram shows how this can operate



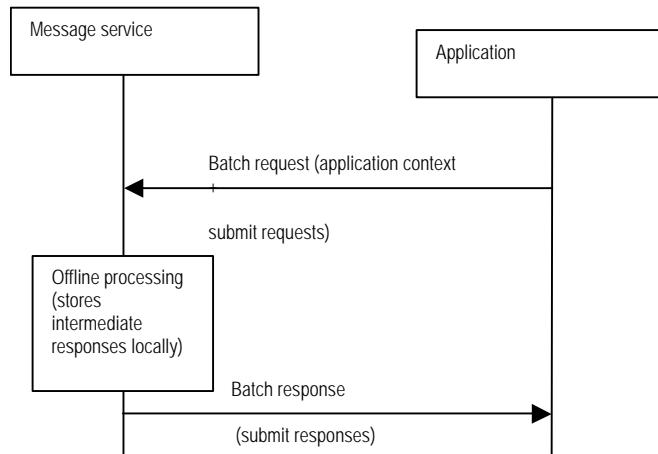
**Figure 4: Asynchronous Session**

### 3.1.4 Batch processing

A batch session allows an application to submit one or more requests for offline processing. This can be used, for example, when submitting a large number of messages or submitting a single message to a large set of recipients.

Each batch request contains a set of MMAP requests or responses and an unbind request. These elements would normally be submitted via an asynchronous bearer protocol such as FTP or SMTP. However, there is nothing to prevent a batch session being initiated over HTTP.

This can be seen in the following diagram:



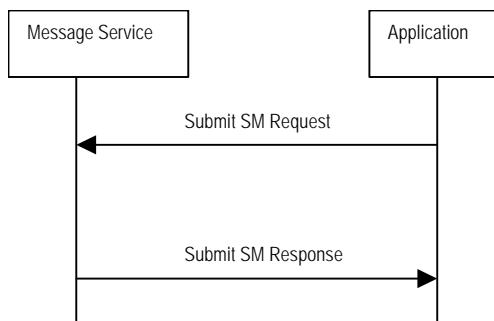
**Figure 5: Batch session**

### 3.2 Message Submission

The message submission procedure allows the application to submit an SMS for delivery to one or more mobile stations (MS). The address of each MS may be specified as part of the submission request or it may be specified in a separate distribution list accessed via an XML link.

#### 3.2.1 Submission to a single destination

When an application submits a message with a single destination specified, the handles the delivery of the messag to the specified destination. Once the message is under the control of the message service, it returns a submission response to the application. This contains a unique identity for the submitted message and can be used for further queries on the message or to track delivery receipts.

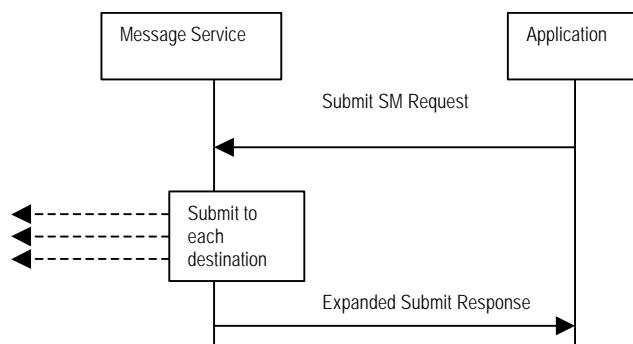


**Figure 6: Submission to a single destination**

### 3.2.2 Submission to multiple destinations

An application can identify multiple destinations for a message. The message service sends a separate message for delivery to each destination on behalf of the application.

The application returns an expanded submission response element when a submission request specifies multiple destinations. This response identifies the destinations to which message creation has failed (e.g. because the destination is invalid) and the destinations to which submission succeeded. For each successful submission the destination address and the associated message identity is returned. For each failed submission, the destination address and the associated error information is returned.

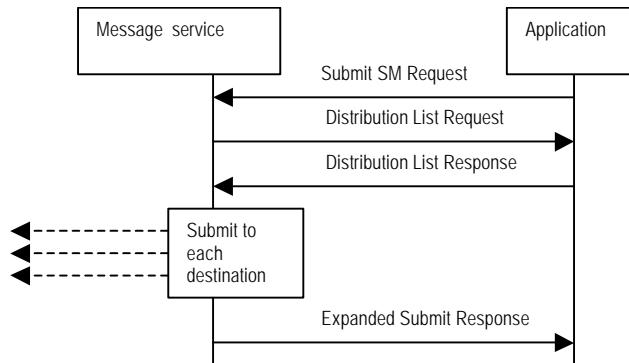


**Figure 7: submission to multiple destinations**

### 3.2.3 Submission using a distribution list

Instead of specifying the destinations within the header of the message, the SM application can specify a link to a distribution list containing the destination addresses. Since this link is a standard XML Xlink attribute it can identify

- ?? An XML structure contained within the request itself (i.e. The destinations are appended to the end of the submission request)
- ?? XML data within the message service itself (i.e. a server-maintained distribution list)
- ?? XML data within the application (i.e. an application-controlled distribution list)

**Figure 8: Distribution List Submission**

### 3.2.3.1 Example

A simple submission request can be encoded as follows

```

<?xml version="1.0" encoding="UTF-8"?>
<mmap:Operation xml:ns:mmap="http://www.smsforum.net/mmap"
  xml:ns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/mmap
    basic_operations.xsd">
  <mmap:ApplicationContext service="testService"
    sessionId="asASF" sourceOperationReference="235" />
  <mmap:SubmitRequest>
    <mmap:Mobi1eMessage>
      <mmap:Header>
        <mmap:Destination>
          <mmap:Number>+44123456</mmap:Number>
        </mmap:Destination>
      </mmap:Header>
      <mmap:Body>
        <mmap:Text>hello world</mmap:Text>
      </mmap:Body>
    </mmap:Mobi1eMessage>
  </mmap:SubmitRequest>
</mmap:Operation>

```

The message service will then return a submission response as follows:

```

<?xml version="1.0" encoding="UTF-8"?>
<mmap:Operation xml:ns:mmap="http://www.smsforum.net/mmap"
  xml:ns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/mmap
    basic_operations.xsd">
  <mmap:ApplicationContext service="testService"
    sessionId="asASF" sourceOperationReference="235" />
  <mmap:SuccessResponse>
    <mmap:MessageRef>4A5678ABFF88</mmap:MessageRef>
  </mmap:SuccessResponse>
</mmap:Operation>

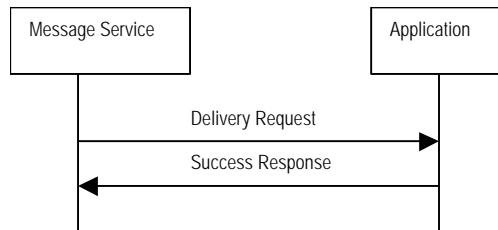
```

## 3.3 Message Delivery

The mobile message service delivers messages and delivery information to applications using a DeliveryRequest element. The application replies with a SuccessResponse element to acknowledge delivery.

The body of a DeliveryRequest contains either a MobileMessage element or else a DeliveryInfo element. This allows the application to distinguish messages that contain delivery information those that contain actual application data and, if required, process them separately.

The following diagram shows the operation of a delivery request.



**Figure 9: Message Delivery**

### 3.3.1 Examples

The message service sends a Message Delivery request to the application in the following format

```

<?xml version="1.0" encoding="UTF-8"?>
<mmap: Operation xmlns:mmap="http://www.smsforum.net/mmap"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/mmap
    basic_operations.xsd">
  <mmap: ApplicationContext service="testService"
    sessionId="asASF" sourceOperationReference="2"/>
  <mmap: DeliverRequest>
    <mmap: MobileMessage>
      <mmap: Header>
        <mmap: Destination>
          <mmap: Application>5555577</mmap: Application>
        </mmap: Destination>
        <mmap: Source>
          <mmap: Number TON="International"
            NPI="ISDN">+44123456</mmap: Number>
          </mmap: Source>
        </mmap: Header>
        <mmap: Body>
          <mmap: Text>hello world</mmap: Text>
        </mmap: Body>
      </mmap: MobileMessage>
    </mmap: DeliverRequest>
  </mmap: Operation>
  
```

The application then sends a success response to the message service with the following format

```

<?xml version="1.0" encoding="UTF-8"?>
<mmap: Operation xml ns: mmap="http://www.smsforum.net/mmap"
  xml ns: xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.smsforum.net/mmap
    basic_operations.xsd">
  <mmap: ApplicationContext service="testService"
  sessionID="asASF" sourceOperationReference="2"/>
  <mmap: SuccessResponse />
</mmap: Operation>

```

### 3.4 Message administration

These facilities allow an application to manage a message after it has been submitted into the network. An application can:

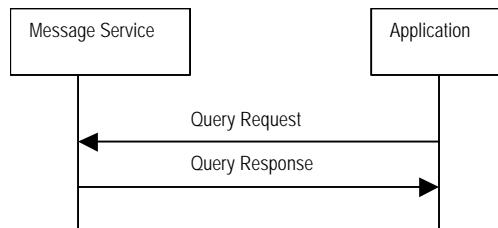
- ?? Query the status of a submitted message
- ?? Modify the contents of a submitted message
- ?? Delete a message that has been submitted but is not yet delivered to its destination.

Since these operations provide direct access to messages within the messaging network, they are defined as privileged operations and gateways may choose to disable them completely or only provide certain applications with access to them.

#### 3.4.1 Querying Message status

An application can query the status of a message within the messaging network. The response indicates whether the message is still in transit or has been delivered to its final destination. If delivered, the response may include the delivery time of the message.

The following diagram shows the operation of a query request.



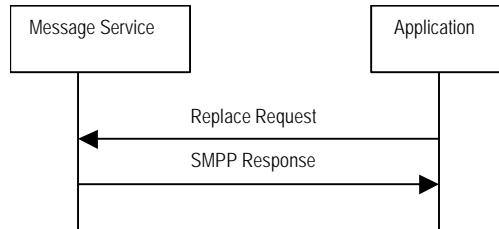
**Figure 10: Query Request**

#### 3.4.2 Modifying a message after submission

An application may modify certain fields of a message after submission if the message is not yet delivered to its final destination. The application can identify the message explicitly through its message identifier or implicitly through specifying the destination address and the service type of the message.

The application can modify the message contents, the scheduled delivery time and the delivery options of the message.

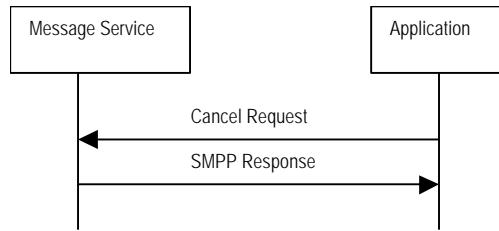
The following diagram shows the operation of a replace request.



### 3.4.3 Cancelling a message after submission

The application can cancel a message that has been submitted to the network, but is not yet delivered to its final destination. The application can identify the message explicitly through its message identifier or implicitly through specifying the destination address and the service type of the message.

The following diagram shows the operation of a cancel request.



## 4 XML Data Elements

### 4.1 XML Schema document structure

This protocol specification is divided into two sets of documents:

- ?? Protocol specifications: These define the basic data elements of the specification.
- ?? Profile specifications: These organize the specification documents so that users can access a subset of protocol functions through a single reference.

#### 4.1.1 Specification documents

The protocol specification is structured into the following set of documents:

Filename	Description
Common.xsd	Defines the root structure of operations, requests and responses. It also defines the basic access control and billing elements.
Session.xsd	Defines elements and datatypes for establishing an

	interactive session.
MessageTransfer.xsd	Defines the requests and responses associated with sending and receiving mobile messages.
MobileMessage.xsd	Defines the basic structure of a mobile message
Addressing.xsd	Defines the addressing elements needed to identify mobile devices and applications.
MessageExtensions.xsd	Defines an extended set of header parameters and body types for a mobile-message.
MessageAdmin.xsd	Defines message administration requests and responses.
Batch.xsd	Defines batch requests and responses

#### 4.1.2 Profile Documents

The following profile documents are defined:

*BasicPush.xsd., more to be added*

### 4.2 Common Definitions

The common module defines a standard set of elements for application access. The root element for all MMAP instance documents is the element “Operation”. An “operation” element contains an application context element followed by either a request or a response. Both request and response elements are defined as being an abstract datatype, i.e. they can never be specified directly in an instance document. Instead, these elements are overwritten by concrete elements derived from these datatypes.

This module defines two standard elements that override Response:

?? SuccessResponse and

?? ErrorResponse:

This module also defines elements for billing. More details on all of these elements is provided the sections following.

#### 4.2.1 ApplicationContext

This element is used to provide a context for an operation. In immediate and batch mode, the ApplicationContext provides a full set of credentials in each operation. In client or peer session mode, the credentials are provided in the session bind procedure and the ApplicationContext element then supplies the session identity for each operation in the session.

This element has the following attributes:

service: This mandatory attribute identifies the service being accessed. It has a type of xs:token.

sessionId: This optional attribute is only specified when request or response is within a session. It identifies the session as managed by the target of the operation except when the operation is a response to a Bind or BackBind request. In these two special cases it identifies the newly-created session on the source entity.

sourceOperationReference: This optional parameter is only specified when the request or response is within a session. It allows the initiator of a request to identifies the associated response within a session. In a response it has the same value as in the associated request. This integer parameter completely under the control of the request initiator and is simply replayed transparently in the operation response.

The ApplicationContext element has one optional child element: AccessControl. This contains an ApplicationIdentity element that identifies the application and an Authentication element currently contains a password element. It is envisioned that these elements may be expanded in the future to contain certificates or other authentication tokens.

```
<mmap: ApplicationContext service="testService">
  <mmap: AccessControl>
    <mmap: ApplicationIdentity>testApp</mmap: ApplicationIdentity>
      <mmap: Authentication>
        <mmap: Password>12345</mmap: Password>
      </mmap: Authentication>
    </mmap: AccessControl>
  </mmap: ApplicationContext>
```

## 4.2.2 Response elements

The “Response” element is used to provide a response to a request. It is overwritten as either a SuccessResponse or an ErrorResponse element.

A SuccessResponse element may contain a message reference

A ErrorResponse element must contain

- o an ErrorCode element that indicates the cause of the error.
- and may also contain
  - o An ErrorDescription element providing a human-readable error description.
  - o A NetworkErrorCode element providing network-specific error codes

### 4.2.2.1 Example

Success response example

```
<mmap: SuccessResponse>
  <mmap: MessageRef>4A5678ABFF88</mmap: MessageRef>
</mmap: SuccessResponse>
```

Error response example

```
<mmap: ErrorResponse>
  <mmap: ErrorCode value="567" />
    <mmap: ErrorDescription>error in XXXX network
el ement</mmap: ErrorDescription>
  <mmap: NetworkErrorCode networkType="IS-95" value="3578"/>
</mmap: ErrorResponse>
```

### 4.2.3 Billing

A “Billing” element contains the following elements:

- ?? ContentCost: This specifies the cost of the submitted message as either a class (e.g. premium), an amount (e.g. 5 dollars) or as a custom value agreed between the application and the gateway.
- ?? BilledService: This identifies the service to which the cost should be applied.
- ?? Account: This specifies a user account to be debited (or credited) for this cost. Note: this could be the application’s own account or an actual user’s account.

#### 4.2.3.1 Example

```
<mmap: Billing id="Billid1">
  <mmap: ContentCost>
    <mmap: CostClass type="Premium" />
  </mmap: ContentCost>
  <mmap: BilledService name="EasyAccess" type="FreeService" />
  <mmap: Account id="Application99" />
</mmap: Billing>
```

## 4.3 Session Management

### 4.3.1 BindRequest

A BindRequest element contains the information needed to establish a session with a gateway. It has no attributes and can have the following child elements:

- ?? SessionControl
- ?? BilledService and
- ?? BindAddress

The **SessionControl** element has one attribute: sessionType. This specifies the type of session being established as either "client" (the default) or "peer". It can contain a FeatureSet element and, optionally, a ReturnSession element.

The **FeatureSet** element is an empty element with the following attributes:

featuresRequired: Those features required in the current session

featuresSupported: The complete set of features supported by the application.

MaxBytesPerMessage: The maximum number of bytes that the application wishes to send in any message in the current session.

The feature attributes have the following type definition:

```
<xs: restriction base="xs:string">
  <xs: enumeration value="MessageSubmission"/>
  <xs: enumeration value="MessageDelivery"/>
  <xs: enumeration value="StatusReporting"/>
  <xs: enumeration value="MessageAdministration"/>
  <xs: enumeration value="ExtendedAddressing"/>
  <xs: enumeration value="ApplicationHeaders"/>
  <xs: enumeration value="NetworkSpecificationSupport"/>
</xs: restriction>
```

If present, the **ReturnSession** element specifies that a return session should be established by the gateway using the service type identified by the gateway and attaching the parameters specified by the application in the body of the element. The application can specify whatever parameters it wishes to be returned by the gateway.

The **BilledService** element specifies the service context required by the application. The application can identify a specific service provided by the gateway and identify its own account for any billing or validation

The **BindAddress** specifies the external address to be used by the gateway in sending messages on behalf of the application and in routing messages to the application. Its format is specified in the Address section.

#### 4.3.1.1 Example

```

<mmap: BindRequest>
<mmap: SessionControl sessionType="peer">
<mmap: FeatureSet
featuresRequired="MessageSubmission"
featuresSupported="MessageSubmission MessageDelivery"/>
<mmap: ReturnSession>
<mmap: SessionReturnPath
link="http: xxx.yyyx.xzzz/app"
service="myservicename" />
<mmap: Parameters>
<mparam>test</mparam>
</mmap: Parameters>
</mmap: ReturnSession>
</mmap: SessionControl>
<mmap: BoundService name="testservice" type="free"/>
<mmap: BindAddress>
<mmap: MAddress>
<mmap: Application>12345</mmap: Application>
</mmap: MAddress>
</mmap: BindAddress>
</mmap: BindRequest>

```

### 4.3.2 BindResponse

A BindResponse element confirms that a session has been established with the server. It also specifies the features supported by the server. It is a subtype of a ResponseType schema type and as such can be identified as a SuccessResponse.

Example

```

<mmap: SuccessResponse xsi:type="mmap: BindResponseType">
<mmap: FeatureSet
featuresSupported="MessageSubmission MessageDelivery
ExtendedAddressing MessageAdministration"
maxBytesPerMessage="140" />
</mmap: SuccessResponse>

```

### 4.3.3 BindBack Request

The BindBack element is used when the gateway wishes to bind a return path for a new peer session.

#### 4.3.3.1 Example

```

<mmap: BindBackRequest sourceService="myservicename">
<mmap: Parameters>
<mparam>test</mparam>
</mmap: Parameters>
</mmap: BindBackRequest>

```

## 4.4 Message Transfer

This operations provided by this module allow an application to send and receive mobile messages. The following top-level elements are defined:

- ?? SubmitRequest
- ?? DeliverRequest
- ?? ExpandedSubmissionResponse

### 4.4.1 Submit

A “SubmitRequest” is used to submit a mobile message or delivery acknowledgement message to a message service. It contains the following child elements:

- ?? Billing (optional)
- ?? MobileMessage or DeliveryInfo (mandatory)

Billing, MobileMessage and DeliveryInfo are specified elsewhere in this document.

### 4.4.2 Deliver

The “DeliverRequest” element is used by the gateway to deliver a mobile message or delivery information to the application. It contains the following mandatory element:

- ?? MobileMessage or DeliveryInfo

These are both described in other sections of this document.

### 4.4.3 Expanded Submission Response

The “ExpandedSubmissionResponse” element is used to deliver a response to a submission request that has been expanded by the gateway into multiple submission requests. This occurs when the submission request contains multiple destinations or links to a distribution list.

This element may contain

- o zero or more SubmitElementError elements that specify the destinations to which the submission failed and give details of the error.
- o Zero or more SubmitElementOK elements providing a message reference and a destination number for each successful submission. The destination number is necessary to qualify the message reference.

#### 4.4.3.1 Example

```

<mmap: ExpandedSubmitResponse>
  <mmap: SubmissionError>
    <mmap: Destination><mmap: Number>+4412345</mmap: Number></mmap: Destination>
      <mmap: ErrorCode value="23"></mmap: ErrorCode>
      </mmap: SubmissionError>
      <mmap: SubmissionOK>
        <mmap: Destination><mmap: Number>+4423478</mmap: Number></mmap: Destination>
        <mmap: MessageRef>4567FFDEE2</mmap: MessageRef>
        </mmap: SubmissionOK>
    </mmap: ExpandedSubmitResponse>
  
```

#### 4.4.4 Delivery Failure Response

A submit operation to a single subscriber will normally result in a standard SuccessResponse or FailureResponse being returned to the caller. In the special case of a submission that involves an immediate message delivery to the subscriber (a “transaction-mode” message), a delivery failure may be returned to the application. This indicates that the message delivery requested in the submission has failed.

The DeliveryErrorResponse element is derived from a standard ErrorResponse datatype and has one extra element:

The DeliveryFailureReason element indicates the reason a mobile message with Application header parameter of transferMode="Immediate" failed to be delivered

### 4.5 Message Administration

This module defines the set of elements that allow an application to manage messages while they are in transit within the network. These are

- o QueryRequest
- o QueryResponse
- o CancelRequest
- o ReplaceRequest

The elements are described in more detail below:

#### 4.5.1 QueryRequest and QueryResponse

QueryRequest is used to query the status of a message after submission. It contains two child elements:

- **MessageRef:** This identifies the message by the identity supplied by the network on submission
- **Source:** This optional element identifies the original source address supplied in the message submission.

QueryResponse contains the result of a successful query operation. It contains the following child elements:

- **MessageRef:** This identifies the message being queried
- **MessageState:** This specifies the current state of the message. This element is also used in the DeliveryInfo element (section 4.6.3)
- **MessageFinalDate:** If present, This element specifies the date at which the processing of the message by the network was completed and message delivery either succeeded or failed.
- **NetworkDeliveryErrorStatus:** This element is present if message delivery failed and a network-specific error value is available.

#### 4.5.1.1 Example

##### Query submission

```
<mmap: QueryRequest>
  <mmap: MessageRef>456789FFFF</mmap: MessageRef>
</mmap: QueryRequest>
```

##### Response

```
<mmap: QueryResponse>
  <mmap: MessageRef>456789FFFF</mmap: MessageRef>
  <mmap: MessageState value="Delivered"/>
</mmap: QueryResponse>
```

#### 4.5.2 CancelRequest

This element is used to cancel a message while it is still in the network. A message can be cancelled either by its service type or by specifying its message identifier explicitly.

#### 4.5.2.1 Example

##### 4.5.2.1.1 Cancel by service type

```
<mmap: Cancel Request>
  <mmap: Cancel ByService serviceType="VoiceMail Alert">
    <mmap: Source>
      <mmap: Number>+44234545</mmap: Number>
    </mmap: Source>
    <mmap: Destination>
      <mmap: Number>+445555</mmap: Number>
    </mmap: Destination>
  </mmap: Cancel ByService>
</mmap: Cancel Request>
```

#### 4.5.2.1.2 Cancel by Reference

```
<mmap: Cancel Request>
  <mmap: Cancel ByReference>
    <mmap: Source>
      <mmap: Number>12345</mmap: Number>
    </mmap: Source>
    <mmap: MessageRef>4577788FF2D</mmap: MessageRef>
  </mmap: Cancel ByReference>
</mmap: Cancel Request>
```

### 4.5.3 ReplaceRequest

This element is used to replace a message before the network delivers it. It contains as child elements the identity of the message to be replaced and a ReplaceMessage element.

The ReplaceMessage element has two child elements;

- o **ReplaceHeader:** This contains a source address that must match that of the original message and, optionally, a DeliveryInfo element.
- o **ReplaceMessage:** This specifies text or binary data to be replaced in the original message.

#### 4.5.3.1 Example

```
<mmap: ReplaceRequest>
  <mmap: MessageRef>458721FFAA1D</mmap: MessageRef>
  <mmap: ReplaceMessage>
    <mmap: ReplaceHeader>
      <mmap: Source>
        <mmap: Number>12345</mmap: Number>
      </mmap: Source>
      <mmap: DeliveryOptions deliveryReceipt="on"/>
    </mmap: ReplaceHeader>
    <mmap: ReplaceBody>
      <mmap: ReplaceText>replacement message</mmap: ReplaceText>
    </mmap: ReplaceBody>
    <mmap: ReplaceMessage>
  </mmap: ReplaceRequest>
```

## 4.6 Mobile Message Module

This module defines a mobile message as a set of XML elements. It can be used to create, manage and transfer messages independently of any devices or protocols. It requires the addressing elements from the Addressing Module.

This module defines two top-level elements

?? MobileMessage and

?? DeliveryInfo

The “MobileMessage” element is defined as having two child elements; “header” and “body”. These will be considered separately in the sections following.

The “DeliveryInfo” element provides a mechanism for signaling status information related to message delivery. It is described separately in a section following.

### 4.6.1 Header

The header element defines the attributes of a mobile message. These are

1. One or more destination addresses or a link to a distribution list
2. Source address
3. Delivery options

Each Destination element is defined as an MMAAddress entity. The contents of this entity were defined in the previous section.

The DistributionListRef element defines an xml link element that links to a set of destination elements. This link can either identify an element in the same document or be the URL of a different XML document containing the actual distribution list.

The DeliveryOptions element defines the scheduling of message delivery and the status information returned as the message delivery proceeds.

#### 4.6.1.1 DeliveryOptions

This element can have the following attributes:

validityPeriod: This specifies the period during which the message is valid in the network. This value is of XML schema type “xs:duration”.

scheduleDeliveryTime: This specifies a time at which delivery should first be attempted by the network. This value is of XML schema type “xs:dateTime”.

deliveryReceipt: This can have one of the following values: “off”, “on”, “FailureOnly”.

readReceipt: This can have one of the following values: "off" or "on".

#### 4.6.1.2 Example

```
<mmap: Header>
  <mmap: Destination>
    <mmap: Number>123456</mmap: Number>
  </mmap: Destination>
  <mmap: DeliveryOptions>
    scheduledDeliveryTime="2001-01-01T09:30:00"
    validityPeriod="P1D"
    deliveryReceipt="on"/>
</mmap: Header>
```

#### 4.6.2 Body

The body of a message contains one of the following:

1. Text
2. XMLContent
3. BinaryData
4. ExternalContent

The Text element contains human-readable text to be sent in the message. This element has an `xml:lang` attribute that defines the language used in the text. This element is encoded according to the language specified in the `xml:lang` attribute and the capabilities of the target destination.

The XMLContent element is a means of passing XML content directly in a message service. The handling of this element is implementation-specific. It could send the content transparently to the destination (e.g. if the content is WML) or it could transcode it in some form (e.g. extract only body text). The default behaviour is to extract and forward the body text of the XML content.

The BinaryContent element allows the user full control of the data in the message. This element contains the hexadecimal encoding of the message user data. The user can specify the encoding directly or accept the default encoding of "binary". The user can also specify the `language` attribute.

The ExternalContent element allows the user to specify a links to external content. It has the following attributes:

- ?? `link` specifies the URL of some external content. This may be a reference to another MiME part in a MiME multi part message or it could be the URL of some external information.
- ?? `format` provides a mechanism for specifying the format of the external content, e.g. by providing the URL of a schema.

Where the message body is a a MiME multipart message, the entire request will be contained in two MiME parts. The first part will contain message a body of "ExternalContent. This ExternalContent element wil hava a "link" attribute that contains a URL of the format [cid://content-identifier](#) where "content-identifier" is the name of the second MiME part. This second MiME part contains the actual encapsulated MiME message that is delivered to the user.

#### 4.6.2.1 Examples

Text

```
<mmap: Body>
  <mmap: Text>hello world</mmap: Text>
</mmap: Body>
```

XML Content

```
<mmap: Body>
  <mmap: XMLContent>
    <myMarkup>
      <test>hello hello</test>
    </myMarkup>
  </mmap: XMLContent>
</mmap: Body>
```

External Content

```
<mmap: Body>
  <mmap: External Content link="http://www.aaa.bbb/xxxxx">
</mmap: Body>
```

Binary Content

```
<mmap: Body>
  <mmap: BinaryData>23FF457F</mmap: BinaryData>
</mmap: Body>
```

#### 4.6.3 Delivery Info

This element contains the information supplied as part of a delivery receipt, a read receipt, a device or user delivery acknowledgement or an intermediate delivery status notification. The "type" attribute of the element defines which of these is being provided.

This body of this element contains the following:

?? Destination

?? Source

?? MessageInfo

?? Text

Of these, only **MessageInfo** is a new element, the others have all been described in the previous section. It provides information about the message through the following elements:

**MessageRef** identifies the message through an network-supplied identifier in its body. It may also supply the original user message reference as an optional attribute.

**MessageState**: identifies the current state of the message. This empty element has a single attribute "Value". This attribute can be set to one of "Enroute", "Delivered", "Expired", "Deleted", "Accepted", "Unknown" or "Rejected".

**NetworkErrorCode** optionally provides network-specific error information.

#### 4.6.3.1 Example

```
<mmap: Del i veryInformati on>
  <mmap: Destinati on>
    <mmap: Number>12345</mmap: Number>
  </mmap: Destinati on>
  <mmap: Source>
    <mmap: Number>5555</mmap: Number>
  </mmap: Source>
  <mmap: MessageInfo>
    <mmap: MessageRef>452377FFAA</mmap: MessageRef>
    <mmap: MessageState value="Deleted"/>
    <mmap: NetworkErrorCode networkType="ANSI- 136"
value="345"/>
  </mmap: MessageInfo>
  <mmap: Text>standard delivery info text</mmap: Text>
</mmap: Del i veryInformati on>
```

## 4.7 Message Extensions Module

The standard MobileMessage module contains of the possible message header and body parameters. The message extensions module provides a set of XML schema definitions that extend the type definitions for Header and Body information.

The header data type is extended with the following extra elements:

- ?? **NetworkHandling**: This element specifies any other network-specific handling of the message.
- ?? **ApplicationOptions**: This element specifies any end-to-end application information
- ?? **DeviceControl**: This element specifies any information directed to the device receiving the message.

The body type is extended to support User Data Headers:

#### 4.7.1 NetworkHandling Element

This element contains a set of information that supports the handling of a message in the network. It has the following attributes:

**serviceType:** This identifies the service provided by the application from a set of the following; “NULL”, “CellularMessaging”, “CellularPaging”, “VoiceMailNotify”, “VoiceMailAlert”, “WAP”, “WAP\_WDP”, “WAP\_WCMP”, “USSD” or “Custom”. If the service is specified as “custom”, the attribute “customServiceName” contains the actual service name.

**storageMode:** This specifies whether or not multiple copies of messages with the same ServiceType parameter should be stored on the network. Possible values are “LatestOnly” and “StoreAll”.

**transferMode:** This specifies how the application would like the message to be transferred through the network. The allowed values are: “Immediate”, “Datagram” and “StoreAndForward”. A message with a transferMode of “immediate” is delivered directly to the subscriber as part of the submission procedure. A message with a transferMode of “Datagram”

**QOSTimeToLive:** This specifies the lifetime in seconds that the application requests the system to keep the message if it cannot be delivered immediately.

Example

```
<mmap: Header xsi:type="mmap: ExtendedHeaderType">
  <mmap: Destination>
    <mmap: Number>+44123456</mmap: Number>
  </mmap: Destination>
  <mmap: NetworkHandling>
    serviceType="CellularPaging"
    storageMode="LatestOnly"
    transferMode="StoreAndForward"
    QOSTimeToLive="1000" />
</mmap: Header>
```

#### 4.7.2 ApplicationOptions element

This element can contain the following optional elements in sequence:

**NumberOfVoiceMails:** This specifies to the number of messages stored in a mailbox.

**Callback:** This specifies a callback number associated with the message. A number of callback elements can be specified within one ApplicationOptions element. This corresponds to a message having multiple callback numbers.

**ITSSession:** This provides the information an ITS application needs to maintain a session with an ITS application on a mobile device.

**ITSReplyType:** This element specifies the ITS reply type to be used by the receiving mobile application in generating a response to this message.

These will now be defined in more detail.

##### 4.7.2.1 NumberOfVoiceMails

This element indicates the number of messages in the subscriber’s voice mailbox. This element is empty and has one mandatory “value” attribute which contains a number in the range 0-99.

#### 4.7.2.2 Callback

A callback element has the following attributes:

- ?? **presentation**: This optional attribute specifies how the callback number is presented to the user on the device. It can have one of the following values: "Allowed", "Restricted" or "NumberNotAvailable".
- ?? **screening**: This optional attribute specifies what screening the network has performed on the callback number. It has one of the following values: "UserProvided-Unscreened", "UserProvided-Verified", "UserProvided-VerificationFailed", "NetworkProvided".
- ?? **numberEncoding**: This mandatory attribute specifies how the callback number is encoded. The possible values are: "DTMF" or "ASCII".

A callback element has the following child elements:

**Number**: This is a mandatory text element containing the callback number.

**Display**: This optional element contains information to be displayed to the user on the device. It has one of the following formats:

```
<Display><Text>display text</Text></Display>
<Display><BinaryData>112277FFAA55</BinaryData></Text>
```

#### 4.7.2.3 ITSSession

The ITSSession element provides application information for a CDMA Interactive Teleservice as defined by the Korean PCS carriers [KORITS]. This empty element has the following attributes:

- ?? **sessionNumber**: This mandatory attribute identifies the session between the application and the mobile device. It is a numeric value in the range 0-255.
- ?? **sequenceNumber**: This mandatory attribute identifies the position of the message in a dialog with the mobile device. It is a numeric value in the range 0-127.
- ?? **finalMessage**: This attribute specifies whether or not the message is the final one of the session. It has a value of "TRUE" or "FALSE". The default value is "FALSE".

#### 4.7.2.4 ITSReply

The ITSReply element provides response information for a CDMA Interactive Teleservice as defined by the Korean PCS carriers [KORITS]. This empty element has one mandatory attribute: "value". This attribute has a number in the range 0-8, where the selected number specifies the type of ITS content in the message.

#### 4.7.2.5 Example

```

<mmap: Header xsi:type="mmap: ExtendedHeaderType">
  <mmap: Destination>
    <mmap: Number>+44123456</mmap: Number>
  </mmap: Destination>
  <mmap: ApplicationOptions>
    <mmap: Callback_numberEncoding="DTMF">
      <mmap: Number>12345</mmap: Number>
    </mmap: Callback>
    <mmap: ITSSession_sessionNumber="5" sequenceNumber="3"/>
  </mmap: ApplicationOptions>
</mmap: Header>

```

### 4.7.3 DeviceControl element

The DeviceControl element has two optional attributes:

- ?? **deviceAckReq**: This specifies whether or not the device should acknowledge receipt of the message. It can have values “on” or “off”.
- ?? **userAckReq**: This specifies whether or not the device should prompt the user to acknowledge receipt of the message. It can have values “on” or “off”.

This element can contain one each of the following child elements (in any order):

- ?? **GSMOptions**: This can specify either GSM Message type 0-7 or full GSM Protocol Identifier(see [GSM 03.40]). It can also specify that the replypath parameter should be set by the network.
- ?? **displayTime**: This specifies a display time for the message to the device.
- ?? **MSValidity**: This provides the device with validity information related to storing the message on the device.
- ?? **MessageWaitInd**: This provides a mechanism whereby the sender can change the message waiting indicator on the mobile device when the message is delivered.
- ?? **ReceiveAlert**: This specifies whether a custom alert should be generated on the mobile device when the message is delivered and can also specify the tone of the alert.

#### 4.7.3.1 GSMOptions

This element has one attribute: replyPath. This can have a value of “Set” or “NotSet”. The default value is “NotSet”.

This element can have one only of the following child elements:

- ?? **messageType**. This specifies the message type component of a GSM Protocol Identifier. A GSM device uses this information to perform a replace on handset operation (i.e. if there is a stored message of this type, replace it with the new message). This element has one mandatory “value” attribute that accepts numbers in the range 0-7.

?? **protocollId**: This allows the application to specify the GSM protocol identifier completely according to [GSM 03.40]. It has one mandatory “value” attribute. This is a hex number comprised of two hex digits.

#### 4.7.3.2 DisplayTime

This empty element has one mandatory “value” attribute. It takes a value of “0”, “1” or “2”. This corresponds to display values of “temporary”, “default” and “invoke” respectively.

#### 4.7.3.3 MSValidity

This element controls the storage of the message on the mobile device. This element is empty and has one mandatory “value” attribute. It can have one of the following values : “Indefinite”, “PowerDown”, “SIDRegistrationArea” or “DisplayOnly”.

#### 4.7.3.4 MessageWaitInd

This element allows the application to control an indication on the mobile device that there are messages waiting for the user. This element is empty and has two attributes:

- ?? **messageType**: this mandatory attribute takes one of the following values: “VoiceMail”, “Fax”, “eMail”, “Other”.
- ?? **setting**: This attribute can have one of the following values: “Active” or “Inactive”. The default value is “Active”.

#### 4.7.3.5 ReceiveAlert

This presence of element indicates to the mobile device that should alert the user in some fashion on message arrival. The optional “tone” attribute contains a numeric value that specifies an alert tone on a TDMA network (see [CMT-136]).

#### 4.7.3.6 Example

```

<mmap: Header xsi:type="mmap: ExtendedHeaderType">
  <mmap: Destination>
    <mmap: Number>+44123456</mmap: Number>
  </mmap: Destination>
  <mmap: DeviceControl>
    <mmap: GSMOptions replyPath="NotSet">
      <mmap: MessageType value="3"/>
    </mmap: GSMOptions>
    <mmap: MessageWaitInd messageType="VoiceMail"
      setting="Inactive"/>
  </mmap: DeviceControl>
</mmap: Header>

```

### 4.8 Address Module

This module defines the addressing formats used to access different Mobile Messaging Entities. The main address datatype “MMAddressType” can have one of a number of child elements:

- ?? Number
- ?? ESMApplication

?? IPAddress

?? Email address

These will each be considered in turn

#### 4.8.1 Number

The element “Number” contains text specifying the address in the form of a telephone number. When no attributes are present the number is taken to be a mobile ISDN number in either national or international format. If the first character of the text is a “+”, the number is taken to be an international number (i.e. “+CCCNNNN” where CCC is the countrycode). Otherwise the number is taken to be a national number and shall be interpreted according to the location of the gateway.

Two attributes can be specified “TON” and “NPI”. These can have the following values:

TON: “Unknown” “International” “National” “Network” “Subscriber” “Alphanumeric” “Abbreviated”

NPI: “Unknown” “ISDN” “Data” “Telex” “LandMobile” “National” “Private” “ERMES” “WAPClient”

##### 4.8.1.1 Examples

```
<Number>+3531234567</Number>
<Number TON="National" NPI="ISDN" >771234567</Number>
<Number TON="Alphanumeric">Voi cemai l</Number>
```

#### 4.8.2 Application

The application element defines the address of an External messaging application.

This can be either a shortcode or an actual application name. When this element is specified as a short code, The gateway is responsible for selecting the appropriate TON and NPI values. Where the application is given as a name, the gateway is presumed to have a standard mechanism for mapping this name onto an appropriate address.

*In the future additional elements may be added to support applications with different addresses in different networks and features such as gateway-assigned addressing (e.g. the application does not care what its address is so long as it is assigned to it for the duration of a bind)*

#### 4.8.3 IP

The IP element defines the IP address. This simple type is a union of the following types: IPV4AddressType and IPV6AddressType. The body contains the IP address in either “aaa.bbb.ccc.ddd” notation or in IPV6 format “A:B:C:D:E:F:G:H”. The only optional attribute is a port number.

#### 4.8.4 EMail

The EMail element defines the RFC2822 address. This simple type is currently defined as a string. This may be further constrained in the future to perform more validity-checking on the address format.

#### 4.8.5 Simple Addressing Examples

```

<mmap: Address>
  <mmap: Number>12345</mmap: Number>
</mmap: Address>
<mmap: Address>
  <mmap: IP port="34">23. 56. 89. 58</mmap: IP>
</mmap: Address>
<mmap: Address>
  <mmap: IP>45. 68. 97. 56: 33: 67: 66: 33</mmap: IP>
</mmap: Address>
<mmap: Address>
  <mmap: Application>TESTAPP</mmap: Application>
</mmap: Address>
<mmap: Address>
  <mmap: Application>
xsi:type="mmap: ApplicationShortCodeType">12345</mmap: Application>
</mmap: Address>
<mmap: Address>
  <mmap: Email>test@smsforum.net</mmap: Email>
</mmap: Address>

```

#### 4.8.6 Extended Addressing

The base MMAddressType can be extended for full control of addressing. The new ExtendedAddress Type has the following extra attributes:

- ?? **networkType**: This attribute can have one of the following values: "Unknown" (default), "GSM", "TDMA", "CDMA", "PDC", "PHS", "iDEN", "AMPS", "Paging".
- ?? **subUnit**: This attribute identifies a sub unit within the device being addressed. It can have one of the following values: "Unknown" (default), "Display", "Equipment", "SmartCard", "ExternalUnit".

An element of this type can have the following extra parameters: Bearer and SubAddress. These are defined in the sections following:

##### 4.8.6.1 Bearer

This element specifies the bearer network to be used in accessing the device. This element is empty and has the following attributes:

Type: This identifies the type of the bearer network. It can have one of the following values: "Unknown" (default), "SMS", "CSD", "Packet", "USSD", "CDPD", "DataTAC", "FLEX", "CellBroadcast".

TelematicsId: This identifies the telematics interworking to be used by the specified bearer network. This is an integer value.

#### 4.8.6.2 SubAddress

This element contains an X.213 subaddress (NSAP) or else a user-specified sub address. The body of the element is a set of hex digits and it has one attribute: addressType that can have a value of "NSAP" (default) or "UserSpecified".

#### 4.8.6.3 Examples

```
<mmap: Destination xsi:type="mmap: ExtendedAddressType">
    <mmap: Number>12345</mmap: Number>
    <mmap: Bearer_telematicsId="34" type="CSD"/>
</mmap: Destination>

<mmap: Destination xsi:type="mmap: ExtendedAddressType"
    networkType="GSM" subUnit="SmartCard">
    <mmap: Number TON="Network" NPI="Tel ex">12345</mmap: Number>
</mmap: Destination>
<mmap: Destination xsi:type="mmap: ExtendedAddressType">
    <mmap: Number>234545</mmap: Number>
    <mmap: SubAddress
addressType="NSAP">34AD23FFFF</mmap: SubAddress>
</mmap: Destination>
```

## 5 Schema Specification

### 5.1 Profiles

#### 5.1.1 BasicPush.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:mmap="http://www.smsforum.net/mmap"
  elementFormDefault="unqualified"
  attributeFormDefault="unqualified">
  <xs:redefine schemaLocation="common.xsd">
    <xs:complexType name="ApplicationContextType">
      <xs:complexContent>
        <xs:restriction
          base="ApplicationContextType">
          <xs:sequence>
            <xs:element name="AccessControl"
              type="AccessControlType" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="service"
            type="xs:token" use="required"/>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>
  </xs:redefine>
  <xs:include schemaLocation="MessageTransfer.xsd"/>
</xs:schema>

```

#### 5.1.2 MMAP.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <xs:include schemaLocation="Session.xsd"/>
  <xs:include schemaLocation="MessageTransfer.xsd"/>
  <xs:include schemaLocation="MessageAdmin.xsd"/>
  <xs:include schemaLocation="MessageExtensions.xsd"/>
</xs:schema>

```

### 5.2 Data definition modules

#### 5.2.1 Common.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xs:attributeFormDefault="unqualified">
    <xs:include schemaLocation="MobileMessage.xsd"/>
    <xs:element name="Operation">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ApplicationContext"
            type="ApplicationContextType"/>
          <xs:choice>
            <xs:element ref="Request"/>
            <xs:element ref="Response"/>
          </xs:choice>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Request" type="RequestType"/>
    <xs:element name="Response" type="ResponseType"/>
    <xs:element name="SuccessResponseType" type="SuccessResponseType"
      substitutionGroup="Response"/>
    <xs:element name="ErrorResponse" type="ErrorResponseType"
      substitutionGroup="Response"/>
    <xs:complexType name="RequestType" abstract="true"/>
    <xs:complexType name="ResponseType" abstract="true"/>
    <xs:complexType name="ApplicationContextType">
      <xs:sequence>
        <xs:element name="AccessControl"
          type="AccessControlType" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="service" type="xs:token"
        use="required"/>
      <xs:attribute name="sourceOperationReference"
        type="xs:unsignedInt"/>
      <xs:attribute name="sessionId" type="xs:NMTOKEN"/>
    </xs:complexType>
    <xs:complexType name="AccessControlType">
      <xs:sequence>
        <xs:element ref="ApplicationIdentity"/>
        <xs:element name="Authentication"
          type="AuthenticationType" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="ApplicationIdentity" type="xs:NMTOKEN"/>
    <xs:complexType name="AuthenticationType">
      <xs:sequence>
        <xs:element ref="Password"/>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="Password" type="xs:string"/>
    <xs:complexType name="SuccessResponseType">
      <xs:complexContent>
        <xs:extension base="ResponseType">
          <xs:sequence>
            <xs:element ref="MessageRef"
              minOccurs="0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:element name="MessageRef" type="MessageRefType"/>
    <xs:complexType name="ErrorResponseType">

```

```

<xs: complexContent>
    <xs: extension base="ResponseType">
        <xs: sequence>
            <xs: element name="ErrorCode"
type="ErrorCodeType"/>
            <xs: element ref="ErrorDescription"
minOccurs="0"/>
            <xs: element name="NetworkErrorCode"
type="NetworkErrorCodeType" minOccurs="0"/>
        </xs: sequence>
    </xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: element name="ErrorDescription" type="xs:string"/>
<xs: complexType name="ErrorCodeType">
    <xs: attribute name="value" type="xs:integer"
use="required"/>
</xs: complexType>
<xs: complexType name="BillingType">
    <xs: sequence>
        <xs: element name="ContentCost"
type="ContentCostType"/>
        <xs: element name="BilledService"
type="BilledServiceType" minOccurs="0"/>
        <xs: element name="Account" type="AccountType"
minOccurs="0"/>
    </xs: sequence>
    <xs: attribute name="identity" type="xs:string"
use="required"/>
</xs: complexType>
<xs: complexType name="AccountType">
    <xs: attribute name="identity" type="xs:token"
use="required"/>
</xs: complexType>
<xs: complexType name="BilledServiceType">
    <xs: attribute name="name" type="xs:NMTOKEN"
use="required"/>
    <xs: attribute name="type" type="xs:NMTOKEN"/>
</xs: complexType>
<xs: complexType name="ContentCostType">
    <xs: choice>
        <xs: element name="CostClass"
type="CostClassType"/>
        <xs: element name="CostAmount"
type="CostAmountType"/>
        <xs: element name="CustomCost"
type="CustomCostType"/>
    </xs: choice>
</xs: complexType>
<xs: complexType name="CostAmountType">
    <xs: attribute name="units" type="xs:string"/>
    <xs: attribute name="amount" type="xs:string"/>
</xs: complexType>
<xs: complexType name="CostClassType">
    <xs: attribute name="type" default="Standard">
        <xs: simpleType>
            <xs: restriction base="xs:NMTOKEN">
                <xs: enumeration value="Free"/>
                <xs: enumeration value="Standard"/>
                <xs: enumeration value="Premium"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="CustomCostType">
    <xs: sequence>

```

```

<xs: element name="CustomParameter"
type="CustomParameterType" maxOccurs="unbounded"/>
  </xs: sequence>
</xs: complexType>
<xs: complexType name="CustomParameterType">
  <xs: attribute name="name" type="xs:string"
use="required"/>
  <xs: attribute name="value" type="xs:string"/>
</xs: complexType>
</xs: schema>

```

## 5.2.2 Session.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <xs:include schemaLocation="common.xsd"/>
  <xs:element name="BindRequest" type="BindRequestType"
    substitutionGroup="Request"/>
    <xs:element name="BindBackRequest" type="BindBackRequestType"
      substitutionGroup="Request"/>
      <xs:element name="UnbindRequest" type="UnbindRequestType"
        substitutionGroup="Request"/>
        <xs:complexType name="BindRequestType">
          <xs:complexContent>
            <xs:extension base="RequestType">
              <xs:sequence>
                <xs:element name="SessionControl" type="SessionControlType"/>
                <xs:element name="BilledService" type="BilledServiceType" minOccurs="0"/>
                <xs:element name="BindAddress" type="BindAddressType" minOccurs="0"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="SessionControlType">
          <xs:sequence>
            <xs:element ref="FeatureSet"/>
            <xs:element name="ReturnSession" type="ReturnSessionInformationType" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="sessionType" default="client">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="client"/>
                <xs:enumeration value="peer"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
        </xs:complexType>
        <xs:element name="FeatureSet">
          <xs:complexType>
            <xs:attribute name="featuresRequired" type="FeatureListType"/>
            <xs:attribute name="featuresSupported" type="FeatureListType"/>

```

```

<xs: attribute name="maxBytesPerMessage"
type="xs: integer"/>
</xs: complexType>
</xs: element>
<xs: simpleType name="FeatureListType">
<xs: list itemType="FeatureType"/>
</xs: simpleType>
<xs: simpleType name="FeatureType">
<xs: restriction base="xs:string">
<xs: enumeration value="MessageSubmission"/>
<xs: enumeration value="MessageDelivery"/>
<xs: enumeration value="StatusReporting"/>
<xs: enumeration value="MessageAdministration"/>
<xs: enumeration value="ExtendedAddressing"/>
<xs: enumeration value="ApplicationHeaders"/>
<xs: enumeration value="NetworkSpecificationSupport"/>
</xs: restriction>
</xs: simpleType>
<xs: complexType name="ReturnSessionInformationType">
<xs: sequence>
<xs: element name="SessionReturnPath">
<xs: complexType>
<xs: attribute name="link"
type="xs:anyURI" use="required"/>
<xs: attribute name="service"
type="xs:token"/>
</xs: complexType>
</xs: element>
<xs: element name="Parameters"
type="ApplicationSuppliedParamsType"/>
</xs: sequence>
</xs: complexType>
<xs: complexType name="BindBackRequestType">
<xs: complexContent>
<xs: extension base="RequestType">
<xs: sequence>
<xs: element name="Parameters"
type="ApplicationSuppliedParamsType"/>
</xs: sequence>
<xs: attribute name="sourceService"
type="xs:token" use="required"/>
</xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: complexType name="ApplicationSuppliedParamsType"
mixed="true">
<xs: sequence maxOccurs="unbounded">
<xs: any processContents="skip"/>
</xs: sequence>
</xs: complexType>
<xs: complexType name="UnbindRequestType">
<xs: complexContent>
<xs: extension base="RequestType"/>
</xs: complexContent>
</xs: complexType>
<xs: complexType name="BindResponseType">
<xs: complexContent>
<xs: extension base="SuccessResponseType">
<xs: sequence>
<xs: element ref="FeatureSet"/>
</xs: sequence>
</xs: extension>
</xs: complexContent>
</xs: complexType>
</xs: schema>

```

### 5.2.3 MessageTransfer.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<x: schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:x="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <x: include schemaLocation="common.xsd"/>
  <x: element name="SubmitRequest" type="SubmitRequestType"
    substitutionGroup="Request"/>
    <x: element name="DeleteRequest" type="DeleteRequestType"
      substitutionGroup="Request"/>
      <x: element name="ExpandedSubmitResponse"
        type="ExpandedSubmitResponseType"
        substitutionGroup="Response"/>
        <x: element name="DeleteErrorResponse"
          type="DeleteErrorResponseType" substitutionGroup="Response"/>
          <x: complexType name="SubmitRequestType">
            <x: complexContent>
              <x: extension base="RequestType">
                <x: sequence>
                  <x: element name="Billing"
                    type="BillingType" minOccurs="0"/>
                  <x: choice>
                    <x: element
                      ref="MobileMessage"/>
                    <x: element
                      ref="DeleteInformation"/>
                    </x: choice>
                  </x: sequence>
                </x: extension>
              </x: complexContent>
            </x: complexType>
            <x: complexType name="DeleteRequestType">
              <x: complexContent>
                <x: extension base="RequestType">
                  <x: choice>
                    <x: element ref="MobileMessage"/>
                    <x: element
                      ref="DeleteInformation"/>
                    </x: choice>
                  </x: extension>
                </x: complexContent>
              </x: complexType>
              <x: complexType name="ExpandedSubmitResponseType">
                <x: complexContent>
                  <x: extension base="SuccessResponseType">
                    <x: sequence>
                      <x: element name="SubmissionError"
                        minOccurs="0" maxOccurs="unbounded"/>
                      <x: complexType>
                        <x: sequence>
                          <x: element
                            ref="Destination"/>
                          <x: element
                            name="ErrorCode" type="ErrorCodeType"/>
                          </x: sequence>
                        </x: complexType>
                      </x: sequence>
                    </x: extension>
                  </x: complexContent>
                </x: complexType>
                <x: complexType name="SubmissionOK"
                  minOccurs="0" maxOccurs="unbounded">
                  <x: complexType>
                    <x: sequence>

```

```

    <xs: element
    ref="Destination"/>
    <xs: element
    ref="MessageRef"/>
        </xs: sequence>
        </xs: complexType>
    </xs: element>
    </xs: sequence>
</xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: complexType name="DeliveryErrorResponseType">
    <xs: complexContent>
        <xs: extension base="ErrorResponseType">
            <xs: sequence>
                <xs: element
name="DeliveryFailureReason" type="DeliveryFailureReasonType"
minOccurs="0"/>
                </xs: sequence>
            </xs: extension>
        </xs: complexContent>
    </xs: complexType>
    <xs: complexType name="DeliveryFailureReasonType">
        <xs: attribute name="value" use="required">
            <xs: simpleType>
                <xs: restriction base="xs:NMTOKEN">
                    <xs: enumeration
value="DestUnavailable"/>
                    <xs: enumeration
value="DestAddressInvalid"/>
                    <xs: enumeration
value="PermNetworkError"/>
                    <xs: enumeration
value="TempNetworkError"/>
                </xs: restriction>
            </xs: simpleType>
        </xs: attribute>
    </xs: complexType>
</xs: schema>

```

#### 5.2.4 MessageAdmin.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
    <xs:include schemaLocation="common.xsd"/>
    <xs:element name="Cancel Request" type="Cancel RequestType"
      substitutionGroup="Request"/>
    <xs:element name="Query Request" type="Query RequestType"
      substitutionGroup="Request"/>
    <xs:element name="Replace Request" type="Replace RequestType"
      substitutionGroup="Request"/>
    <xs:element name="Query Response" type="Query ResponseType"
      substitutionGroup="Response"/>
    <xs:complexType name="Cancel RequestType">
        <xs:complexContent>
            <xs:extension base="RequestType">
                <xs:choice>

```

```

                <xs: element name="Cancel ByReference"
type="Cancel ByReferenceType"/>
                <xs: element name="Cancel ByService"
type="Cancel ByServiceType"/>
                </xs: choice>
            </xs: complexContent>
        </xs: complexType>
<xs: complexType name="Cancel ByReferenceType">
    <xs: sequence>
        <xs: element ref="Source"/>
        <xs: element name="MessageRef"
type="MessageRefType"/>
    </xs: sequence>
</xs: complexType>
<xs: complexType name="Cancel ByServiceType">
    <xs: sequence>
        <xs: element ref="Source"/>
        <xs: element ref="Destination"/>
    </xs: sequence>
    <xs: attribute name="serviceType" type="xs:string"
use="required"/>
</xs: complexType>
<xs: complexType name="MessageFinalDateType">
    <xs: attribute name="value" type="xs:string"
use="required"/>
</xs: complexType>
<xs: complexType name="NetworkDeliveryErrorStatusType">
    <xs: attribute name="value" type="xs:string"
use="required"/>
</xs: complexType>
<xs: complexType name="QueryRequestType">
    <xs: complexContent>
        <xs: extension base="RequestType">
            <xs: sequence>
                <xs: element name="MessageRef"
type="MessageRefType"/>
                <xs: element ref="Source"
minOccurs="0"/>
            </xs: sequence>
        </xs: extension>
    </xs: complexContent>
</xs: complexType>
<xs: complexType name="QueryResponseType">
    <xs: complexContent>
        <xs: extension base="SuccessResponseType">
            <xs: sequence>
                <xs: element name="MessageRef"
type="MessageRefType"/>
                <xs: element name="MessageState"
type="MessageStateType"/>
                <xs: element name="MessageFinalDate"
type="MessageFinalDateType" minOccurs="0"/>
                <xs: element
name="NetworkDeliveryErrorStatus"
type="NetworkDeliveryErrorStatusType" minOccurs="0"/>
            </xs: sequence>
        </xs: extension>
    </xs: complexContent>
</xs: complexType>
<xs: complexType name="ReplaceRequestType">
    <xs: complexContent>
        <xs: extension base="RequestType">
            <xs: sequence>
                <xs: element name="MessageRef"
type="MessageRefType"/>

```

```

<xs: element name="ReplaceMessage"
type="ReplaceMessageType"/>
</xs: sequence>
</xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: element name="ReplaceBinaryData" type="xs:string"/>
<xs: complexType name="ReplaceBodyType">
<xs: choice>
<xs: element ref="ReplaceText"/>
<xs: element ref="ReplaceBinaryData"/>
</xs: choice>
</xs: complexType>
<xs: complexType name="ReplaceHeaderType">
<xs: sequence>
<xs: element ref="Source"/>
<xs: element name="DeliveryOptions"
type="DeliveryOptionsType" minOccurs="0"/>
</xs: sequence>
</xs: complexType>
<xs: complexType name="ReplaceMessageType">
<xs: sequence>
<xs: element name="ReplaceHeader"
type="ReplaceHeaderType"/>
<xs: element name="ReplaceBody"
type="ReplaceBodyType"/>
</xs: sequence>
</xs: complexType>
<xs: element name="ReplaceText" type="xs:string"/>
</xs: schema>

```

### 5.2.5 Addressing.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smsforum.net/mmap"
  xmlns:mmap="http://www.smsforum.net/mmap"
  xmlns="http://www.smsforum.net/mmap"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xs:complexType name="MAddressType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="Number"
type="NumberType"/>
        <xs:element name="Application"
type="ApplicationAddressType"/>
        <xs:element name="IP" type="IPAddressType"/>
        <xs:element name="Email"
type="RFC2822AddressType"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="NumberType">
    <xs:annotation>
      <xs:documentation>Phone number format SME
address</xs:documentation>
      <xs:annotation>
        <xs:simpleContent>
          <xs:extension base="xs:token">
            <xs:attribute name="TON"
default="International">
              <xs:simpleType>

```

```

<xs: restriction
base="xs: NMTOKEN">
    <xs: enumeration>
    val ue="Unknown"/>
    <xs: enumeration>
    val ue="International"/>
    <xs: enumeration>
    val ue="National"/>
    <xs: enumeration>
    val ue="Network"/>
    <xs: enumeration>
    val ue="Subscriber"/>
    <xs: enumeration>
    val ue="Alphanumeric"/>
    <xs: enumeration>
    val ue="Abbreviated"/>
        </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
    <xs: attribute name="NPI" default="ISDN">
        <xs: simpleType>
            <xs: restriction
base="xs: NMTOKEN">
    <xs: enumeration>
    val ue="Unknown"/>
    <xs: enumeration>
    val ue="ISDN"/>
    <xs: enumeration>
    val ue="Data"/>
    <xs: enumeration>
    val ue="Tel ex"/>
    <xs: enumeration>
    val ue="LandMobile"/>
    <xs: enumeration>
    val ue="National"/>
    <xs: enumeration>
    val ue="Private"/>
    <xs: enumeration>
    val ue="ERMES"/>
    <xs: enumeration>
    val ue="WAPClient"/>
        </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
    </xs: extension>
    </xs: simpleContent>
</xs: complexType>
<xs: complexType name="BearerType">
    <xs: attribute name="type" default="Unknown">
        <xs: simpleType>
            <xs: restriction base="xs: NMTOKEN">
                <xs: enumeration value="Unknown"/>
                <xs: enumeration value="SMS"/>
                <xs: enumeration value="CSD"/>
                <xs: enumeration value="Packet"/>
                <xs: enumeration value="USSD"/>
                <xs: enumeration value="CDPD"/>
                <xs: enumeration value="DataTAC"/>
                <xs: enumeration value="FLEX"/>
            <xs: enumeration/>
    </xs: simpleType>
    </xs: restriction>
    </xs: attribute>
    <xs: attribute name="teleioticsId" type="xs: integer"/>
</xs: complexType>
<xs: simpleType name="ApplicationShortCodeType">

```

```

<xs: restriction base="ApplicationAddressType">
    <xs: pattern value="(\+)?([0-9])+" />
</xs: restriction>
</xs: simpleType>
<xs: simpleType name="ApplicationNameType">
    <xs: restriction base="ApplicationAddressType">
        <xs: pattern value="(\+)?([0-9])+" />
    </xs: restriction>
</xs: simpleType>
<xs: simpleType name="ApplicationAddressType">
    <xs: restriction base="xs:string"/>
</xs: simpleType>
<xs: complexType name="ExtendedAddressType">
    <xs: complexContent>
        <xs: extension base="MAddressType">
            <xs: sequence>
                <xs: element name="Bearer" type="BearerType" minOccurs="0"/>
                <xs: element name="SubAddress" type="SubAddressType" minOccurs="0"/>
            </xs: sequence>
            <xs: attribute name="networkType" default="Unknown">
                <xs: simpleType>
                    <xs: restriction>
                        <xs: enumeration>
                            <xs: value value="NMTOKEN"/>
                            <xs: value value="Unknown"/>
                            <xs: value value="GSM"/>
                            <xs: value value="TDMA"/>
                            <xs: value value="CDMA"/>
                            <xs: value value="PDC"/>
                            <xs: value value="PHS"/>
                            <xs: value value="iDEN"/>
                            <xs: value value="AMPS"/>
                            <xs: value value=" Paging"/>
                        </xs: enumeration>
                    </xs: restriction>
                </xs: simpleType>
            </xs: attribute>
        </xs: sequence>
    </xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: attribute name="subUnit" default="Unknown">
    <xs: simpleType>
        <xs: restriction>
            <xs: enumeration>
                <xs: value value="NMTOKEN"/>
                <xs: value value="Unknown"/>
                <xs: value value="Display"/>
                <xs: value value="Equipment"/>
                <xs: value value="SmartCard"/>
                <xs: value value="ExternalUnit"/>
            </xs: enumeration>
        </xs: restriction>
    </xs: simpleType>
</xs: attribute>
</xs: complexType>

```

```

</xs: complexType>
<xs: simpleType name="IPCombi nedAddressType">
    <xs: union memberTypes="IPV4AddressType
IPV6AddressType"/>
    </xs: simpleType>
    <xs: complexType name="IPAddressType">
        <xs: simpleContent>
            <xs: extension base="IPCombi nedAddressType">
                <xs: attribute name="port"
type="xs: integer"/>
            </xs: extension>
        </xs: simpleContent>
    </xs: complexType>
    <xs: simpleType name="SubAddressBufferType">
        <xs: restriction base="xs: string">
            <xs: pattern value="[0-9a-fA-f]{0,44}" />
        </xs: restriction>
    </xs: simpleType>
    <xs: complexType name="SubAddressType">
        <xs: annotation>
            <xs: documentation>contains an NSAP or user-
specified subaddress in hex format. An odd number of hex digits
is allowed.</xs: documentation>
            </xs: annotation>
        <xs: simpleContent>
            <xs: extension base="SubAddressBufferType">
                <xs: attribute name="addressType"
default="NSAP">
                    <xs: simpleType>
                        <xs: restriction
base="xs: NMTOKEN">
                            <xs: enumeration
value="NSAP" />
                            <xs: enumeration
value="UserSpecified" />
                            <xs: restriction>
                                <xs: simpleType>
                                    <xs: attribute>
                                        <xs: extension>
                                            <xs: simpleContent>
                                                <xs: attribute>
                                                    <xs: extension>
                                                        <xs: simpleType>
                                                            <xs: restriction
source="http://www.ietf.org/rfc/rfc2396.txt"
xml:lang="en">SOURCE: IETF "RFC 2396: Uniform Resource
Identifiers (URI): Generic Syntax" [1998-08]
</xs: documentation>
                                                </xs: annotation>
                                                <xs: restriction base="xs: string">
                                                    <xs: pattern value="(([0-9]{1,2})|[1][0-9]{2})|[2][0-
4][0-9]|([2][5][0-5])\.)\.{3}([0-9]{1,2})|[1][0-9]{2}|[2][0-4][0-
9]|([2][5][0-5])" />
                                                </xs: restriction>
                                            </xs: simpleType>
                                            <xs: simpleType name="IPV6AddressType">
                                                <xs: annotation>
                                                    <xs: documentation
source="http://www.ietf.org/rfc/rfc2373.txt"
xml:lang="en">SOURCE: IETF "RFC 2373: IP Version 6 Addressing
Architecture" [1998-7]
                                                </xs: documentation>
                                                </xs: annotation>
                                                <xs: restriction base="xs: string">
                                                    <xs: pattern value="([A-Fa-f0-9]{1,4}):){7}[A-Fa-f0-
9]{1,4}" />
                                                </xs: restriction>
                                            </xs: simpleType>
                                        </xs: extension>
                                    </xs: attribute>
                                </xs: simpleType>
                            </xs: extension>
                        </xs: attribute>
                    </xs: simpleType>
                </xs: restriction>
            </xs: simpleContent>
        </xs: annotation>
    </xs: complexType>
</xs: simpleType>

```

```

</xs: simpleType>
<xs: simpleType name="RFC2822AddressType">
    <xs: restriction base="xs:string"/>
</xs: simpleType>
<xs: element name="MAddress" type="MAddressType"/>
<xs: element name="Destination" type="MAddressType"/>
<xs: element name="Source" type="MAddressType"/>
<xs: element name="ExtendedMAddress"
type="ExtendedAddressType"/>
<xs: group name="DestinationGroup">
    <xs: choice>
        <xs: element ref="Destination"/>
        <xs: element name="Destinations">
            <xs: complexType>
                <xs: sequence maxOccurs="unbounded">
                    <xs: element ref="Destination"/>
                </xs: sequence>
            </xs: complexType>
        </xs: element>
        <xs: element name="DestinationList"
type="URLType"/>
    </xs: choice>
</xs: group>
<xs: complexType name="BindAddressType">
    <xs: choice>
        <xs: element ref="MAddress"/>
        <xs: element name="Range">
            <xs: complexType>
                <xs: simpleContent>
                    <xs: restriction
base="NumberType">
                        <xs: pattern value="[0-9A-Z^$]"/>
                    </xs: restriction>
                </xs: simpleContent>
            </xs: complexType>
        </xs: element>
    </xs: choice>
</xs: complexType>
<xs: complexType name="URLType">
    <xs: attribute name="link" type="xs:anyURI"/>
</xs: complexType>
</xs: schema>

```

### 5.2.6 MessageExtensions.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs: schema targetNamespace="http://www.smsforum.net/mmap"
xml ns: xs="http://www.w3.org/2001/XMLSchema"
xml ns="http://www.smsforum.net/mmap"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs: include schemaLocation="Addressing.xsd"/>
    <xs: include schemaLocation="MobileMessage.xsd"/>
    <xs: import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="xml.xsd"/>
    <xs: complexType name="ExtendedHeaderType">
        <xs: complexContent>
            <xs: extension base="MessageHeaderType">
                <xs: sequence>
                    <xs: element name="NetworkHandling"
type="NetworkHandlingType" minOccurs="0"/>

```

```

<xs: element name="ApplicationOptions"
type="ApplicationOptionsType" minOccurs="0"/>
<xs: element name="DeviceControl"
type="DeviceControlType" minOccurs="0"/>
</xs: sequence>
</xs: extension>
</xs: complexContent>
</xs: complexType>
<xs: complexType name="ExtendedDeliveryInfoType">
<xs: complexContent>
<xs: extension base="DeliveryInfoType">
<xs: sequence>
<xs: element name="UserResponseCode"
type="UserResponseType"/>
</xs: sequence>
<xs: attribute name="InfoType"
use="required">
<xs: simpleType>
<xs: restriction>
base="xs:NMTOKEN">
<xs: enumeration>
val ue="NetworkReceipt"/>
<xs: enumeration>
val ue="ReadReceipt"/>
<xs: enumeration>
val ue="DeviceAck"/>
<xs: enumeration>
val ue="UserAck"/>
<xs: enumeration>
val ue="IntermediateStatusNotification"/>
<xs: restriction>
</xs: simpleType>
</xs: attribute>
</xs: extension>
</xs: complexContent>
</xs: complexType name="NetworkHandlingType">
<xs: attribute name="serviceType" use="optional"
default="NULL">
<xs: simpleType>
<xs: restriction base="xs:NMTOKEN">
<xs: enumeration value="NULL"/>
<xs: enumeration>
val ue="CellularMessaging"/>
<xs: enumeration>
val ue="Cellular Paging"/>
<xs: enumeration>
val ue="VoiceMailNotify"/>
<xs: enumeration>
val ue="VoiceMailAlert"/>
<xs: enumeration value="WAP"/>
<xs: enumeration value="WAP_WDP"/>
<xs: enumeration value="WAP_WCMP"/>
<xs: enumeration value="USSD"/>
<xs: enumeration value="Custom"/>
</xs: restriction>
</xs: simpleType>
</xs: attribute>
<xs: attribute name="customServiceName" type="xs:string"
use="optional"/>
<xs: attribute name="storageMode" use="optional"
default="StoreAll">
<xs: simpleType>
<xs: restriction base="xs:NMTOKEN">
<xs: enumeration value="LatestOnly"/>
<xs: enumeration value="StoreAll"/>
</xs: restriction>

```

```

        </xs: simpleType>
    </xs: attribute>
    <xs: attribute name="transferMode" use="optional">
        <xs: simpleType>
            <xs: restriction base="xs:NMTOKEN">
                <xs: enumeration value="Immediate"/>
                <xs: enumeration value="Datagram"/>
            <xs: enumeration
value="StoreAndForward"/>
                </xs: restriction>
            </xs: simpleType>
        </xs: attribute>
        <xs: attribute name="QOSTimeToLive"
type="xs:nonNegativeInteger"/>
        <xs: complexType>
            <xs: complexType name="ApplicationOptionsType">
                <xs: sequence>
                    <xs: element name="NumberOfVoiceMails"
type="NumberOfMessagesType" minOccurs="0"/>
                    <xs: element name="Callback" type="CallbackType"
minOccurs="0" maxOccurs="unbounded"/>
                    <xs: element name="ITSSession"
type="ITSSessionType" minOccurs="0"/>
                    <xs: element name="ITSReplyType"
type="ITSReplyTypeType" minOccurs="0"/>
                </xs: sequence>
            </xs: complexType>
            <xs: complexType name="DeviceControlType">
                <xs: all>
                    <xs: element name="GSMOptions"
type="GSMOptionsType" minOccurs="0"/>
                    <xs: element name="DisplayTime"
type="DisplayTimeType" minOccurs="0"/>
                    <xs: element name="MSValidity"
type="MSValidityType" minOccurs="0"/>
                    <xs: element name="MessageWaitInd"
type="MessageWaitIndType" minOccurs="0"/>
                    <xs: element name="ReceiveAlert"
type="ReceiveAlertType" minOccurs="0"/>
                </xs: all>
                <xs: attribute name="deviceAckReq" use="optional">
                    <xs: simpleType>
                        <xs: restriction base="xs:NMTOKEN">
                            <xs: enumeration value="off"/>
                            <xs: enumeration value="on"/>
                        </xs: restriction>
                    </xs: simpleType>
                </xs: attribute>
                <xs: attribute name="userAckReq" use="optional">
                    <xs: simpleType>
                        <xs: restriction base="xs:NMTOKEN">
                            <xs: enumeration value="off"/>
                            <xs: enumeration value="on"/>
                        </xs: restriction>
                    </xs: simpleType>
                </xs: attribute>
            </xs: complexType>
            <xs: complexType name="GSMOptionsType">
                <xs: sequence>
                    <xs: choice>
                        <xs: element name="ProtocolId">
                            <xs: complexType>
                                <xs: attribute name="value">
                                    <xs: simpleType>
                                        <xs: restriction
base="xs:hexBinary">

```

```

value="2" />                                         <xs: length
                                                 </xs: restriction>
                                                 </xs: simpleType>
                                                 </xs: attribute>
                                                 </xs: complexType>
</xs: element>                                         </xs: element>
<xs: element name="MessageType">                      <xs: attribute name="value">
                                                 <xs: complexType>
                                                 <xs: attribute name="value">
                                                 <xs: simpleType>
                                                 <xs: restriction>
base="xs: integer">                                     </xs: restriction>
                                                 </xs: simpleType>
                                                 </xs: attribute>
                                                 </xs: complexType>
                                                 </xs: element>
                                                 </xs: choice>
</xs: sequence>                                         </xs: sequence>
<xs: attribute name="replyPath" default="NotSet">       <xs: restriction base="xs: NMTOKEN">
                                                 <xs: enumeration value="Set"/>
                                                 <xs: enumeration value="NotSet"/>
                                                 </xs: restriction>
                                                 </xs: simpleType>
                                                 </xs: attribute>
</xs: complexType>                                       </xs: complexType>
<xs: complexType name="CallbackType">                     <xs: sequence>
                                                 <xs: element name="Number" type="NumberType"/>
                                                 <xs: element name="Display" type="StringType"/>
                                                 <xs: attribute name="m0n0ccurs" value="0"/>
                                                 </xs: sequence>
                                                 <xs: attribute name="presentation">
                                                 <xs: simpleType>
                                                 <xs: restriction base="xs: NMTOKEN">
                                                 <xs: enumeration value="Allowed"/>
                                                 <xs: enumeration value="Restricted"/>
                                                 <xs: enumeration value="Unrestricted"/>
val ue="NumberNotAvailable"/>                           </xs: restriction>
                                                 </xs: simpleType>
                                                 </xs: attribute>
                                                 <xs: attribute name="screening">
                                                 <xs: simpleType>
                                                 <xs: restriction base="xs: NMTOKEN">
                                                 <xs: enumeration value="UserProvided-  
Unscreened"/>
                                                 <xs: enumeration value="UserProvided-  
Verified"/>
                                                 <xs: enumeration value="UserProvided-  
VerificationFailed"/>
                                                 <xs: enumeration value="NetworkProvided"/>
val ue="NetworkProvided"/>                             </xs: enumeration>
                                                 </xs: simpleType>
                                                 </xs: attribute>
                                                 <xs: attribute name="numberEncoding" use="required">
                                                 <xs: simpleType>
                                                 <xs: restriction base="xs: NMTOKEN">
                                                 <xs: enumeration value="DTMF"/>

```

```

                <xs: enumeration value="ASCII" />
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="RegisteredDeliveryType">
<xs: complexType name="CallbackDisplayType">
    <xs: choice>
        <xs: element name="Text" type="MessageTextType"/>
        <xs: element name="BinaryData" type="BinaryDataType"/>
    </xs: choice>
</xs: complexType>
<xs: complexType name="ITSSessionType">
    <xs: attribute name="sessionNumber" use="required">
        <xs: simpleType>
            <xs: restriction base="xs: integer">
                <xs: minValue value="0"/>
                <xs: maxValue value="255"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
    <xs: attribute name="sequenceNumber" use="required">
        <xs: simpleType>
            <xs: restriction base="xs: integer">
                <xs: minValue value="0"/>
                <xs: maxValue value="127"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
    <xs: attribute name="finalMessage" default="FALSE">
        <xs: simpleType>
            <xs: restriction base="xs: NMTOKEN">
                <xs: enumeration value="TRUE"/>
                <xs: enumeration value="FALSE"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="ITSReplyTypeType">
    <xs: attribute name="value" use="required">
        <xs: simpleType>
            <xs: restriction base="xs: integer">
                <xs: minValue value="0"/>
                <xs: maxValue value="8"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="UserResponseCodeType">
    <xs: attribute name="value" type="xs:string" use="required"/>
</xs: complexType>
<xs: complexType name="DisplayTimeType">
    <xs: attribute name="value" default="1">
        <xs: simpleType>
            <xs: restriction base="xs: NMTOKEN">
                <xs: enumeration value="0"/>
                <xs: enumeration value="1"/>
                <xs: enumeration value="2"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="MSValidityType">
    <xs: attribute name="value" default="Indefinite">
        <xs: simpleType>

```

```

<xs: restriction base="xs:NMTOKEN">
    <xs: enumeration value="Indefinite"/>
    <xs: enumeration value="PowerDown"/>
    <xs: enumeration
value="SIDRegistrationArea"/>
        <xs: enumeration value="Displayonly"/>
    </xs: restriction>
</xs: simpleType>
</xs: attribute>
<xs: complexType name="MessageWaitIndType">
    <xs: attribute name="setting" default="Active">
        <xs: simpleType>
            <xs: restriction base="xs:NMTOKEN">
                <xs: enumeration value="Active"/>
                <xs: enumeration value="Inactive"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
    <xs: attribute name="messageType" use="required">
        <xs: simpleType>
            <xs: restriction base="xs:NMTOKEN">
                <xs: enumeration value="VoiceMail"/>
                <xs: enumeration value="Fax"/>
                <xs: enumeration value="eMAIL"/>
                <xs: enumeration value="Other"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="NumberOfMessagesType">
    <xs: attribute name="value" use="required">
        <xs: simpleType>
            <xs: restriction base="xs:integer">
                <xs: minInclusive value="0"/>
                <xs: maxInclusive value="99"/>
            </xs: restriction>
        </xs: simpleType>
    </xs: attribute>
</xs: complexType>
<xs: complexType name="ReceiveAlertType">
    <xs: attribute name="tone" type="xs:integer"
use="optional"/>
</xs: complexType>
<xs: complexType name="ExtendedBodyType">
    <xs: complexContent>
        <xs: extension base="MessageBodyType">
            <xs: sequence>
                <xs: element name="UserDataHeader"
type="UserDataHeaderType" minOccurs="0"/>
            </xs: sequence>
            <xs: attribute name="charEncoding">
                <xs: simpleType>
                    <xs: restriction
base="xs:NMTOKEN">
                        <xs: enumeration
value="DefaultAlphabet"/>
                        <xs: enumeration
value="IA5"/>
                        <xs: enumeration
value="OCTET"/>
                        <xs: enumeration
value="Latin1"/>
                        <xs: enumeration
value="JIS"/>
                        <xs: enumeration
value="Cyrillic"/>

```

```
value="Latin-Hebrew" />           <xs: enumeration>
value="UCS2" />                  <xs: enumeration>
value="Pictogram" />             <xs: enumeration>
value="ISO-2022-JP" />            <xs: enumeration>
value="Extended-Kanji-JIS" />    <xs: enumeration>
value="KCS-C-5601" />              <xs: enumeration>
                                </xs: restriction>
                                </xs: simpleType>
                                </xs: attribute>
                                <xs: attribute ref="xml:lang" />
                            </xs: extension>
                            </xs: complexContent>
                        </xs: complexType>
                        <xs: complexType name="UDHElementType">
                            <xs: attribute name="ident" type="xs: hexBinary" use="required" />
                            <xs: attribute name="value" type="xs: hexBinary" use="required" />
                        </xs: complexType>
                        <xs: complexType name="UserDataHeaderType">
                            <xs: choice>
                                <xs: element ref="EncodedUDH" />
                                <xs: element name="UDHElement" type="UDHElementType" maxOccurs="unbounded" />
                            </xs: choice>
                        </xs: complexType>
                        <xs: element name="EncodedUDH" type="xs: hexBinary" />
                    </xs: schema>
```