[MS-OMS]:

Office Mobile Service Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- No Trade Secrets. Microsoft does not claim any trade secret rights in this documentation.
- Patents. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplq@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments	
7/13/2009	0.1	Major	Initial Availability	
8/28/2009	0.2	Editorial	Revised and edited the technical content	
11/6/2009	0.3	Editorial	Revised and edited the technical content	
2/19/2010	1.0	Editorial	Revised and edited the technical content	
3/31/2010	1.01	Editorial	Revised and edited the technical content	
4/30/2010	1.02	Editorial	Revised and edited the technical content	
6/7/2010	1.03	Editorial	Revised and edited the technical content	
6/29/2010	1.04	Editorial	Changed language and formatting in the technical content.	
7/23/2010	1.05	Major	Significantly changed the technical content.	
9/27/2010	1.05	None	No changes to the meaning, language, or formatting of the technical content.	
11/15/2010	1.05	None	No changes to the meaning, language, or formatting of the technical content.	
12/17/2010	1.05	None	No changes to the meaning, language, or formatting of the technical content.	
3/18/2011	1.05	None	No changes to the meaning, language, or formatting of the technical content.	
6/10/2011	1.05	None	No changes to the meaning, language, or formatting of the technical content.	
1/20/2012	2.0	Major	Significantly changed the technical content.	
4/11/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.	
7/16/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.	
9/12/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.	
10/8/2012	2.1	Minor	Clarified the meaning of the technical content.	
2/11/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.	
7/30/2013	2.2	Minor	Clarified the meaning of the technical content.	
11/18/2013	2.2	None	No changes to the meaning, language, or formatting of the technical content.	
2/10/2014	2.2	None	No changes to the meaning, language, or formatting of the technical content.	
4/30/2014	2.2	None	No changes to the meaning, language, or formatting of the technical content.	

Date	Revision History	Revision Class	Comments	
7/31/2014	2.2	None	No changes to the meaning, language, or formatting of the technical content.	
10/30/2014	2.2	None	No changes to the meaning, language, or formatting of the technical content.	
3/16/2015	3.0	Major	Significantly changed the technical content.	
6/23/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.	

Table of Contents

1		n				
	1.1 Glossary					
		nces				
		rmative References				
		ormative References				
		ew				
		es				
	1.3.1.1	Protocol Server				
	1.3.1.2	Protocol Clients				
		enarios 1				
	1.3.2.1	Obtaining Information from the Protocol Server 1				
	1.3.2.2	Obtaining Information from an Authenticated User 1				
	1.3.2.3	Data Communication				
		nship to Other Protocols				
		uisites/Preconditions				
		bility Statement				
		ning and Capability Negotiation				
		-Extensible Fields				
	1.9 Standa	rds Assignments 1	2			
2	Messages	1	3			
_		ort				
		on Message Syntax 1				
		mespaces				
		ssages				
	2.2.2.1	Mobile message packaged as MIME formatted e-mail message				
	2.2.2.1.1					
	2.2.2.1	•				
	2.2.2.1					
	2.2.2.1					
	2.2.2.1	1.4 From	4			
	2.2.2.1					
	2.2.2.1.2	Message Body1	5			
	2.2.2.1		5			
	2.2.2.1		5			
	2.2.3 Ele	ments 1	5			
	2.2.4 Cor	mplex Types1				
	2.2.4.1	tAudio	7			
	2.2.4.2	tBody 1				
	2.2.4.3	tContent				
	2.2.4.4	tDeliveryError1				
	2.2.4.5	tHeader 2	_			
	2.2.4.6	tImg2	1			
	2.2.4.7	tLayout2				
	2.2.4.8	tMeta 2				
	2.2.4.9	tMmsSlides				
	2.2.4.10	tPar				
	2.2.4.11	tRegion2				
	2.2.4.12	tRoot-layout2				
	2.2.4.13	tText				
	2.2.4.14	tTo				
	2.2.4.15	tUser				
	2.2.4.16	tXmsBody				
	2.2.4.17	tXmsData2				
	2.2.4.18	tXmsHeader 2	5			

	2.2.4.19		esponse	
			es	
	2.2.5.1		edServiceTypeEnum	
	2.2.6.1			
	2.2.7 G	roups		27
			roups	
			ata Structures	
3	Protocol D	etails		28
3.				
			ata Model	
			n	
			ocessing Events and Sequencing Rules	
	3.1.4.1		viceInfo	
	3.1.4.1.		ssages	
	_	.1.1.1	GetServiceInfoSoapIn	
		.1.1.2	GetServiceInfoSoapOut	
	3.1.4.1.		ments	
		.1.2.1	GetServiceInfo	
		.1.2.2	GetServiceInfoResponse	
	3.1.4.1.	.1.2.3	serviceInfonplex Types	
	3.1.4.1.		tServiceInfo	
		.1.3.1	tSupportedService	
	3.1.4.		tsms_sender	
	3.1.4.		tLONG_SMS_SENDER	
	3.1.4.		tMMS_SENDER	
	3.1.4.1.		nple Types	
	_	1.4.1	tAuthenticationTypeEnum	
	3.1.4.1.	.5 Attı	ributes	
	3.1.4.1.		oups	
	3.1.4.1.	.7 Attı	ribute Groups	34
	3.1.4.2	GetUse	rInfo	34
	3.1.4.2.	.1 Mes	ssages	
	3.1.4.	.2.1.1	GetUserInfoSoapIn	
	3.1.4.	.2.1.2	GetUserInfoSoapOut	
	3.1.4.2.	.2 Eler	ments	
	_	.2.2.1	GetUserInfo	
	_	.2.2.2	GetUserInfoResponse	
		.2.2.3	xmsUser	
	3.1.4.		userInfo	
	3.1.4.2.		mplex Types	
	3.1.4.		tXmsUser	
	3.1.4.		tUserInfo	
	3.1.4.		tUserError	
	3.1.4.2. 3.1.4.2.		nple Types	
	3.1.4.2.		ributes	
	3.1.4.2.		ribute Groups	
	3.1.4.2.		Xms	
	3.1.4.3		ssages	
		.1 Mes .3.1.1	DeliverXmsSoapIn	
		.3.1.1	DeliverXmsSoapOut	
	3.1.4.3.		ments	
	3.1.4.3.		DeliverXms	
		.3.2.2	DeliverXmsResponse	
	J.1.T.		2 c t c. x 3 k c spori de minimum minimu	40

		244222	_						4.0
		3.1.4.3.2.3		ta					
	2	3.1.4.3.2.4 .1.4.3.3		sponse ypes					
		.1.4.3.3	Simple Typ	ypes es			 		40
		.1.4.3.5							
		.1.4.3.6							
		.1.4.3.7		roups					
				ch					
		.1.4.4.1							
	5	3.1.4.4.1.1	Deliver	XmsBatchS	nanIn		 		41 41
		3.1.4.4.1.2		XmsBatchSo					
	3	.1.4.4.2							
		3.1.4.4.2.1		XmsBatch					
		3.1.4.4.2.2		XmsBatchRe					
		3.1.4.4.2.3		tch					
		3.1.4.4.2.4		sponses					
	3	.1.4.4.3		ypes					
		3.1.4.4.3.1		atch					
		3.1.4.4.3.2	tXmsDa	ataInBatch			 		43
		3.1.4.4.3.3	tXmsRe	esponseWith	ıId		 		44
	3	.1.4.4.4	Simple Typ	es			 		44
	3	.1.4.4.5	Attributes.				 		44
	_	.1.4.4.6							
		.1.4.4.7		roups					
				ssage to clie					
	3.1.5								
_	3.1.6								
3	.2			- I					
	3.2.1			el					
	3.2.2 3.2.3								
	3.2.4								
	3.2.4			g Events and					
	3.2.5								
4	Proto	ocol Examp							
-	.1								
-	.2								
	.3								
-	.4	DeliverXms	Batch				 		49
4	.5			om Server t					
		from Recipi	ent's Phone	:			 		50
5	Secu	rity					 		51
	.1			s for Implen					
	.2			neters					
			•						
6	Appe	ndix A: Ful	ı WSDL	•••••		•••••	 	•••••	52
7	Appe	ndix B: Pro	duct Beha	vior			 		59
0		ge Trackin							
8	cnan	уе ггаскій	y	•••••		•••••	 	•••••	60
9	Inde	x					 		61

1 Introduction

The Office Mobile Service Protocol specifies the protocol used to transmit mobile messages between a protocol client and a protocol server.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

- alert: A message that is passed to a protocol client to notify it when specific criteria are met.
- **ASCII**: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.
- **authenticated user**: A built-in security group specified in [MS-WSO] whose members include all users that can be authenticated by a computer.
- **authentication**: The act of proving an identity to a server while providing key material that binds the identity to subsequent communications.
- **certificate**: A certificate is a collection of attributes (1) and extensions that can be stored persistently. The set of attributes in a certificate can vary depending on the intended usage of the certificate. A certificate securely binds a public key to the entity that holds the corresponding private key. A certificate is commonly used for **authentication** and secure exchange of information on open networks, such as the Internet, extranets, and intranets. Certificates are digitally signed by the issuing certification authority (CA) and can be issued for a user, a computer, or a service. The most widely accepted format for certificates is defined by the ITU-T X.509 version 3 international standards. For more information about attributes and extensions, see [RFC3280] and [X509] sections 7 and 8.
- **Hypertext Markup Language (HTML)**: An application of the Standard Generalized Markup Language (SGML) that uses tags to mark elements in a document, as described in [HTML].
- **language code identifier (LCID)**: A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.
- **Multipurpose Internet Mail Extensions (MIME)**: A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], and [RFC2047].
- **Simple Mail Transfer Protocol (SMTP)**: A member of the TCP/IP suite of protocols that is used to transport Internet messages, as described in [RFC5321].
- **site**: A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as web site.
- **SOAP action**: The HTTP request header field used to indicate the intent of the SOAP request, using a **URI** value. See [SOAP1.1] section 6.1.1 for more information.
- **SOAP body**: A container for the payload data being delivered by a SOAP message to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

- **SOAP fault**: A container for error and status information within a SOAP message. See [SOAP1.2-1/2007] section 5.4 for more information.
- **Synchronized Multimedia Integration Language (SMIL)**: An XML-based language that enables a data stream to be divided, transmitted as separate streams, and then recombined as a single stream, as described in [W3C-SMIL3.0].
- **Uniform Resource Identifier (URI)**: A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].
- **Uniform Resource Locator (URL)**: A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].
- web service: A unit of application logic that provides data and services to other applications and can be called by using standard Internet transport protocols such as HTTP, Simple Mail Transfer Protocol (SMTP), or File Transfer Protocol (FTP). Web services can perform functions that range from simple requests to complicated business processes.
- **XML fragment**: Lines of text that adhere to XML tag rules, as described in [XML], but do not have a Document Type Definition (DTD) or schema, processing instructions, or any other header information.
- **XML namespace**: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].
- **XML namespace prefix**: An abbreviated form of an **XML namespace**, as described in [XML].
- **XML schema**: A description of a type of XML document that is typically expressed in terms of constraints on the structure and content of documents of that type, in addition to the basic syntax constraints that are imposed by XML itself. An XML schema provides a view of a document type at a relatively high level of abstraction.
- MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[GIF89a] CompuServe Incorporated, "Graphics Interchange Format(sm)", Graphics Interchange Format Programming Reference, July 1990, http://www.w3.org/Graphics/GIF/spec-gif89a.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008, http://rfc-editor.org/rfc/rfc5321.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

1.2.2 Informative References

[IANA-CharSets] IANA, "Character Sets", Last Updated 2010-11-04, http://www.iana.org/assignments/character-sets

[MS-LCID] Microsoft Corporation, "Windows Language Code Identifier (LCID) Reference".

1.3 Overview

This protocol allows a client to remotely send mobile messages to a server that processes and delivers these messages to a recipient's phone. The protocol of data communication from a protocol server to a phone is not in the scope of this document.

A typical scenario for using this protocol is a data communication application between software in a computer with a phone. The software, as a protocol client, sends the data as specified in this document to a protocol server, and the protocol server will deliver the data to the phone. The phone can reply to the protocol server and the protocol server delivers the message to the protocol client via **SMTP** protocol.

Another scenario for using this protocol is an **alert** application sent from software in a computer to a phone. The software, as a protocol client, sends the data as specified in this document to a protocol server, and the protocol server will deliver the data to the phone. Such an application could use this protocol to provide user a notification on the phone when user has no access to the computer or Internet.

1.3.1 Roles

Two roles are always being played whenever this protocol is used. There is always a protocol client issuing request to a protocol server, and there is always a protocol server to receive, process and respond to the requests of protocol clients.

1.3.1.1 Protocol Server

The protocol server implements the **Web service** described by this protocol. It also maintains the database of **authenticated users** who can send a valid request to this server, as well as delivers the data sent from these users to the recipients' phones according to the request. It also collects the replies from the phones and delivers to the protocol clients accordingly.

1.3.1.2 Protocol Clients

Protocol clients issue commands to the protocol server via the Web service methods described in this protocol.

The client is required to implement the message delivery mechanism from client to server.

The client can also accept replies to the messages from a server, if two-way communication is required between the client and the recipient's phone. If the application does not require a reply (such as an alert application), the client need not implement the interpretation mechanism in receiving the reply message.

1.3.2 Scenarios

The methods described by this service enable several types of data communication operations.

1.3.2.1 Obtaining Information from the Protocol Server

Protocol clients can send a request to understand what the protocol server can offer. A common usage is for the protocol clients to configure the behavior of itself according to the server's information.

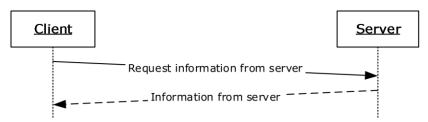


Figure 1: Obtaining information from a protocol server

1.3.2.2 Obtaining Information from an Authenticated User

Protocol clients can obtain more detailed information from an authenticated user from the server. A common usage is for the protocol clients to obtain user's phone number as the recipient of an alert.

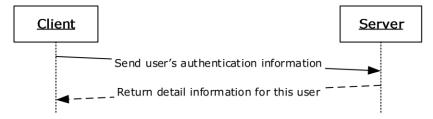


Figure 2: Obtaining information from an authenticated user

Prior to the initiation of this request, the client is configured with the user's authenticated information.

1.3.2.3 Data Communication

Protocol clients can initiate communications with the protocol server by sending a **SOAP** message. The protocol server will send a response to the client. If the protocol server receives a mobile message as a reply, this will be delivered to the protocol client via **SMTP**.

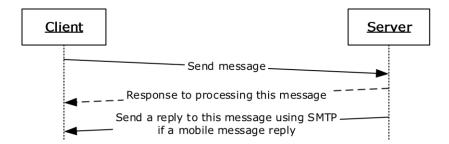


Figure 3: Data communication

Prior to the initiation of sending messages to server, the client is configured with the user's authenticated information.

1.4 Relationship to Other Protocols

This protocol uses the SOAP message protocol for formatting request and response messages, as described in [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages by using HTTPS, as described in [RFC2818].

The following diagram shows the underlying messaging and transport stack used by the protocol:

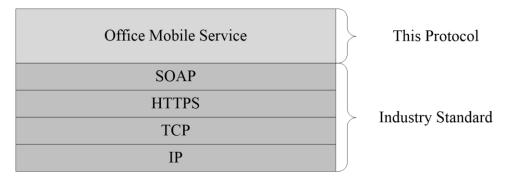


Figure 4: This protocol in relation to other protocols

This protocol has no interactions with parallel protocols, nor are there other protocols that substitute for it.

1.5 Prerequisites/Preconditions

This protocol operates against a **site** that is identified by a **URL** that is known by protocol clients. The protocol client also knows the user's authentication information for sending a request to retrieve user information or deliver message to the server. The protocol requires that SOAP data transferring under HTTPS to protect the user's authentication information.

The protocol server maintains records of known protocol clients. For each client the server will store the information needed to authenticate messages sent from the client, and the e-mail addresses needed to deliver messages to the client.

1.6 Applicability Statement

The protocol is designed to allow protocol clients to send mobile messages to the protocol server.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL might specify differences that reflect actual Microsoft product behavior. For example, the schema definition might allow for an element to be **empty**, **null**, or **not present** but the behavior of the protocol as specified restricts the same elements to being **non-empty**, **not null**, and **present**.

2.1 Transport

Protocol servers MUST support SOAP over HTTPS, as specified in [RFC2818], for securing communication with clients.

Protocol messages, including service discovery and mobile messages from protocol client to protocol server MUST be formatted as specified either in [SOAP1.1] section 4 or in [SOAP1.2/1] section 5. Protocol server faults MUST be returned either using HTTP status codes as specified in [RFC2616] section 10 or using **SOAP faults** as specified either in [SOAP1.1] section 4.4 or in [SOAP1.2/1] section 5.4.

The replies of mobile messages sent from protocol servers to protocol clients MUST be transmitted using Simple Mail Transfer Protocol (SMTP), as specified in [RFC5321].

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses **XML schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL, as specified in [WSDL].

2.2.1 Namespaces

This specification defines and references various **XML namespaces** using the mechanisms specified in [XMLNS]. Although this specification associates a specific **XML namespace prefix** for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[WSDL]
tns	http://schemas.microsoft.com/office/Outlook/2006/OMS	
S	http://www.w3.org/2001/XMLSchema	[WSDL]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[WSDL]
(none) http://schemas.microsoft.com/office/Outlook/2006/OMS		
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

2.2.2 Messages

The WSDL message definitions are specified in their respective operations in the section 3.1.

The following message definition is applied to the reply messages sent from server to client after the server collects this reply message from the recipient's phone.

2.2.2.1 Mobile message packaged as MIME formatted e-mail message

The following sections describe the structure of mobile messages packaged as **Multipurpose**

Internet Mail Extensions (MIME) formatted e-mail messages.

2.2.2.1.1 Message Headers

When encoding a reply into an e-mail message, set SMTP headers as described in the following sections so that the client can properly recognize the incoming e-mail messages as coming from a mobile phone.

2.2.2.1.1.1 Content-Class

Sets the message header "Content-class" to one of following values:

Text message content class: MS-OMS-SMS

Use this if the mobile message is a text message.

Multimedia message content class: MS-OMS-MMS

Use this if the mobile message is a multimedia message.

2.2.2.1.1.2 X-MS-Reply-To-Mobile

Adds the following header specifically for conveying the recipient's mobile number:

X-MS-Reply-To-Mobile: The value of this header SHOULD be a valid mobile phone number.

2.2.2.1.1.3 To

The value of the **To** field is an e-mail address provided by the authenticated user for receiving incoming mobile messages.

2.2.2.1.1.4 From

The value of the **From** field is the e-mail address that is used for sending the reply. The server is required to provide a unique SMTP address to the mobile recipient for sending back replies.

2.2.2.1.1.5 Subject

If the reply e-mail message is for an incoming text message, the value of the **Subject** field MUST be either the first 40 characters of the text message body or the first line of the text message (if there are multiple lines in the message body).

If the reply e-mail message is for an incoming multimedia message, the server MUST set the subject of the e-mail message as the subject of the multimedia message. The subject of the MMS message

14 / 64

SHOULD remind users to view the message content by opening the message as shown in the following

example:

Subject of MMS Message (open the message to view content).

2.2.2.1.2 Message Body

2.2.2.1.2.1 Incoming Text Message

To compose the message body for an incoming text message, the server MUST create a plain text

MIME part for the text message content by adding the following headers:

Content-Type: text/plain; charset=xxxx

Content-Transfer-Encoding: quoted-printable

Examples of valid charset values are "us-ascii" (ASCII) and "gb2312" (Simplified Chinese). The server can also use multipart/alternative content type and provide a HTML view of the message body.

A reference of valid **charset** values can be found in [IANA-CharSets].

2.2.2.1.2.2 Incoming Multimedia Message

When composing the message body for incoming multimedia messages, the server MUST encode multimedia messages as MIME-formatted SMTP mails.

If **SMIL** is available, the server MUST compose the MIME body as multipart/related:

Content-Type: multipart/related; type="application/smil";

The first MIME part of the SMIL file MUST be as follows:

Content-Type: application/smil; name="mmspresent.smil"

Media parts of the MMS message MUST be encoded as MIME parts with corresponding media types following the SMIL file.

If SMIL is not available, the server MUST compose the MIME body as multipart/mixed:

Content-type: multipart/mixed

The server MUST encode media parts of the multimedia message as MIME parts with the

corresponding media types.

2.2.3 Elements

This specification does not define any common XML schema element definitions.

15 / 64

2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions defined by this specification. XML schema complex type definitions that are specific to a particular operation are described with the operation.

:Audio		
	Base64 encoded audio object.	
Body	Body part of SMIL.	
tContent .	Content of the xmsBody .	
:DeliveryError	Indicates the success or failure of the attempt to send this message to the intended recipient.	
:Header	Header part of SMIL.	
tImg	Base64 encoded image object.	
tLayout	Layout part of SMIL.	
:Meta	Metadata indicating the author of the SMIL.	
t MmsSlides The presentation description part for multimedia messages.		
:Par	Specifies multimedia message's slides.	
Region	Specifies the region of the text or image.	
tRoot-layout	Specifies phone screen resolution, in pixels, and background color.	
:Text	Plain text object.	
то	One or more recipients of this mobile message.	
:User	The user information of the sender of this mobile message.	
tXmsBody	Content body of this mobile message.	
Content of mobile message.		
tXmsHeader Header information of this mobile message.		

This complex type contains one or more error elements that indicate the success or failure of the attempt to send this message to the intended recipient.

2.2.4.1 tAudio

Base64 encoded audio object.

src: An identifier that refers to the contentId attribute of the content element.

2.2.4.2 tBody

Body part of SMIL.

par: Multiple occurrences. Specifies multimedia message slides; MUST be an **XML fragment** that conforms to the XML schema of the **tPar** complex type.

2.2.4.3 tContent

Content of the xmsBody.

contentType: MIME content type.

Addition to the content, text messages support the "text/plain" content type. The media objects listed in following table for multimedia messages are supported.

Content	MIME type	Description
Text	text/plain	Plain text. Can be used by both text and multimedia messages.

Content	MIME type	Description
Static image.	image/jpeg	96 DPI. Smaller, or equal to, the mobile screen size defined in the root-layout element of the xmsData string. Base64 encoded. Only applies to multimedia messages.
Multi-frame image.	image/gif	[GIF89a], 96 DPI, maximum of 256 colors. Smaller or equal to the mobile screen size defined in the root-layout element of the xmsData string. Base64 encoded. Only applies to multimedia messages.
MIDI audio format.	audio/mid	MIDI format. Base64 encoded. Only applies to multimedia messages.
AMR sound format.	audio/AMR	AMR format, single channel, 8K Hz. Base64 encoded. Only applies to multimedia messages.

contentId: Content identifier referred in SMIL body part. The server MUST ignore it for text message and non-slide mode multimedia message.

contentLocation: Indicates the file name of a media object, which can be used as the default file name when the object is saved.

2.2.4.4 tDeliveryError

Indicates the success or failure of the attempt to send this message to the intended recipient.

The error element has two child elements: **content**, which is a string containing a description or parameters of the error, and **recipientList**, which is a string containing a semi-colon-delimited list of recipients that are affected by the error.

At most, one content element and one recipientList element can be defined for each error element. The absence of a recipientList element means that the error applies to all recipients.

Each error code has two required attributes: code and severity. The possible values are as follows:

- When either a complete and successful send, or the service is sending a non-error message to the client:
 - Code: The "ok" value MUST be set.
 - **Severity:** The "neutral" value MUST be set.
- When the message has not been delivered to one or more recipients:
 - **Code:** The error code MUST be set follow the different situations in the following table or define its own protocol.

• **Severity:** The "failure" value MUST be set.

Value of code attribute	Explanation	content child element	recipientList child element
invalidUser	Invalid or unrecognized user identifier or password.	Not applicable	Not applicable
unregisteredUser	User has not registered for the service. The service provider returns "invalidUser" if it cannot make the judgment based on the user identifier.	Not applicable	Not applicable
unregisteredService	User has not subscribed to the service listed in the content element.	"SMS" or "MMS"	Not applicable
expiredUser	User's prepayment is used up or the registration is expired. The error code is for the service provider who provides prepaid service.	Not applicable	Not applicable
invalidRecipient	One or more recipients are not valid or not recognized. Returns semicolon delimited recipients in the recipientList element.	Not applicable	Recipient1; Recipient2;
crossCarrier	One or more recipients are from a carrier that is not supported by the sender's carrier. Returns semicolon delimited recipients in the < recipientList> element.	Not applicable	Recipient1; Recipient2;
invalidChar	Message subject or body contains characters or words that are not allowed by local policy or not supported by the service provider.	Not applicable	Not applicable
invalidMedia	Invalid or unsupported media. Returns content IDs of invalid media in the content element. (Attribute only applies to MMS messages).	location1; location2	Not applicable
perDayMsgLimit	Exceeded limit on the number of messages a user can send per day. Returns the limit number in the content element.	Limit on number of messages per day in the form: "# SMS" or "# MMS"	Recipient1; Recipient2;
perMonthMsgLimit	Exceeded limit on the number of messages a user can send per month. Returns the limit number in the content element.	Limit on number of messages per month in the form: "# SMS" or "# MMS"	Recipient1; Recipient2;
lengthLimit	Exceeded length limit of SMS message. Returns maximum	DB or mixed limit; SB	Not applicable

Value of code attribute	Explanation	content child element	recipientList child element
	length limits for single byte messages and double byte messages in the content element.	limit	
sizeLimit	Exceeded size limit of MMS message.	Maximum size allowed (in bytes) of MMS messages	Not applicable
slidesLimit	Exceeded limit on number of slides an MMS message can have.	Maximum number of slides allowed per MMS message	Not applicable
invalidFormat	Invalid or unrecognized XMS data format.	Not applicable	Not applicable
serviceNetwork	Service-side network problem, for example, unable to connect to short message service center (SMSC).	Not applicable	Recipient1; Recipient2;
noScheduled	Scheduled send is not supported. The message was sent immediately.	Not applicable	Not applicable
lowBalance	User's account balance is low. Returns current balance and cost per message in the content element.	Returns current balance and cost per message separated by semi-colons (use currency symbol before each number). For example, "\$5.00;\$0.10"	Recipient1; Recipient2;
ceasedService	Notifies client that the service is terminated.	Not applicable	Not applicable
other	Use this error code for all other errors.	Error message	Recipient1; Recipient2;

- When the server information or service properties are changed, the server SHOULD return the following xmsResponse element:
 - Code: The "serviceUpdate" value MUST be set.
 - **Severity:** The "neutral" value MUST be set.
 - Content: UTC time in format of YYYY-MM-DDThh:mm:ssZ MUST be set. Second part ("ss")
 MUST be "00" and the accuracy is at minute level.

The following are notes on the use of the **tDeliveryError** type and **Severity** attribute:

- If the error element is not included in an xmsResponse string, the client assumes that a failure error occurred.
- If the error element without the code attribute is included in an **xmsResponse** string, the client assumes that an unknown failure error occurred.

- If the error element without the severity attribute is included in an **xmsResponse** string, the client assumes that the severity is "neutral."
- If multiple error codes are returned, the error that has the highest severity attribute ("failure" is higher than "neutral") decides whether the client generates a Non-delivery Report (NDR) and sends it to the user. If there are one or more "failure" errors, the OMS client generates an NDR letting the user know that the message has not been delivered.

2.2.4.5 tHeader

Header part of SMIL.

meta: Metadata indicating the author of the SMIL; MUST be an XML fragment that conforms to the XML schema of the **tMeta** complex type.

layout: Layout part of SMIL; MUST be an XML fragment that conforms to the XML schema of the **tLayout** complex type.

2.2.4.6 tImg

Base64 encoded image object.

```
<xs:complexType name="tImg">
        <xs:attribute name="src" type="xs:string" use="required" />
        <xs:attribute name="region" type="xs:string" use="required" />
</xs:complexType>
```

src: An identifier that refers to the **contentId** attribute of the **content** element.

region: Region for displaying the image. It has the same value as the **id** attribute of the **tRegion** which is a child of the **tLayout** complex type. The value could be "Image" or "Text".

2.2.4.7 tLayout

Layout part of SMIL.

root-layout: Specifies phone screen resolution, in pixels, and background color. MUST be an XML fragment that conforms to the XML schema of the **tRoot-layout** complex type.

region: Specifies the region of the text or image. MUST be an XML fragment that conforms to the XML schema of the **tRegion** complex type.

2.2.4.8 tMeta

Metadata indicating the author of the SMIL.

name: The server MUST ignore this value.

content: The server MUST ignore this value.

2.2.4.9 tMmsSlides

The presentation description part for multimedia messages.

head: Header part of SMIL; MUST be an XML fragment that conforms to the XML schema of the **tHeader** complex type.

body: Body part of SMIL; MUST be an XML fragment that conforms to the XML schema of the **tBody** complex type.

2.2.4.10 tPar

Specifies the slide of the multimedia message.

img: Base64 encoded image object; MUST be an XML fragment that conforms to the XML schema of the **tImg** complex type.

text: Plain text object; MUST be an XML fragment that conforms to the XML schema of the **tText** complex type.

audio: Base64 encoded audio object; MUST be an XML fragment that conforms to the XML schema of the **tAudio** complex type.

dur: Specifies duration in milliseconds that the slide will be played.

2.2.4.11 tRegion

Specifies the region of the text or image.

```
<xs:complexType name="tRegion">
    <xs:attribute name="id" type="xs:string" use="required" />
    <xs:attribute name="left" type="xs:string" use="required" />
    <xs:attribute name="top" type="xs:string" use="required" />
    <xs:attribute name="width" type="xs:string" use="required" />
    <xs:attribute name="height" type="xs:string" use="required" />
    <xs:attribute name="fit" type="xs:string" use="required" />
    <xs:attribute name="fit" type="xs:string" use="required" /></xs:complexType>
```

id: MUST be either "Image" or "Text" indicating the type of region being defined.

left: Specifies the left position as a percentage of the region relative to the left edge of the phone screen.

top: Specifies the top position as a percentage of the region relative to the top edge of the phone screen.

width: Specifies the width of the region as a percentage. The width of the region is equal to the actual value of the width of the region, divided by the actual value of the width of the window.

height: Specifies the height of the region as a percentage.

fit: Must be either "slice" or "stream", to indicate the MMS data communication mode. The width of the region is equal to the actual value of the width of the region, divided by the actual value of the width of the window.

2.2.4.12 tRoot-layout

Specifies phone screen resolution, in pixels, and background color.

width: Phone screen's width in pixels.

height: Phone screen's height in pixels.

background-color: The background color of slides. The background color is a RGB color encoded as a string with format "#RRGGBB". Each of **RR**, **GG**, and **BB** is a hexadecimal encoded number ranging from 00 for 0 to "FF" for 255. **RR** represents the red value. **GG** represents the green value. **BB** represents the blue value.

2.2.4.13 tText

Plain text object.

src: An identifier that refers to the **contentId** attribute of the **content** element.

region: Region for displaying the text. It has the same value as the **id** attribute of the **tRegion**, which is a child of **tLayout** complex type. The value could be "Image" or "Text".

foreground-color: Foreground color of the text. The foreground color is an RGB color encoded as a string with format "#RRGGBB". Each of RR, GG and BB is a hexadecimal encoded number ranging from "00" for 0 to "FF" for 255. RR represents the red value, GG represents the green value, and BB represents the blue value.

2.2.4.14 tTo

One or more recipients of this mobile message.

recipient: Multiple occurrences. Refers to the recipient's mobile phone number (address). At least one recipient is required.

2.2.4.15 tUser

The user information of the sender of this mobile message.

userId: User's identification of the sender.

password: User's password of the sender.

replyPhone: Reply number or callback number. Service providers that do not support callback numbers can ignore this element.

customData: Protocol server MUST ignore this value.

2.2.4.16 tXmsBody

Content body of this mobile message.

mmsSlides: The presentation description part for multimedia messages; MUST be an XML fragment that conforms to the XML schema of the **tMmsSlides** complex type.

Content: Represents one of the following:

- The text messages, if format attribute of xmsBody is "SMS". Multiple content elements are
 possible for text message with each element representing a division of a longer message. The
 server MUST deliver each of the elements as individual text messages in sequence.
- Text, image, or audio object for a slide if format attribute of xmsBody is "MMS" and SMIL part is available (slide mode).
- Page of text, image, or audio object of multimedia message, if format attribute of xmsBody is "MMS" and SMIL part is not available (non-slide mode).

MUST be an XML fragment that conforms to the XML schema of the **tContent** complex type.

format: Specifies the format or type of the xmsBody element. It MUST be "SMS" for a text message or "MMS" for a multimedia message.

2.2.4.17 tXmsData

Content of a mobile message.

user: The user information of the sender of this mobile message; MUST be an XML fragment that conforms to the XML schema of the **tUser** complex type.

xmsHead: Header information of this mobile message; MUST be an XML fragment that conforms to the XML schema of the **tXmsHeader** complex type.

xmsBody: Content body of this mobile message; MUST be an XML fragment that conforms to the XML schema of the **tXmsBody** complex type.

client: See section 2.2.6.1.

2.2.4.18 tXmsHeader

Header information of this mobile message.

scheduled: Indicates that the message is to be sent at a specified time. UTC time in the format YYYY-MM-DDThh:mm:ssZ. The accuracy is at the minute level, so the second element ("ss") MUST be "00".

The server needs to convert the scheduled time to the local time zone. If the specified time is in the past, the message is sent immediately.

requiredService: The delivery service required to delivery this mobile message; MUST be an XML fragment that conforms to the XML schema of the **tRequiredServiceTypeEnum** simple type.

sourceType: Indicates that the message is generated from which type of client, or a specific feature of a client.

to: One or more recipients of this mobile message; MUST be an XML fragment that conforms to the XML schema of the **tTo** complex type.

subject: Subject of the message. For a text message, the server MUST ignore the subject. If left unspecified for a multimedia message, the server MAY return error or send the multimedia message out without a subject.

2.2.4.19 tXmsResponse

This complex type contains one or more error elements that indicate the success or failure of the attempt to send this message to the intended recipient.

error: It MUST be an XML fragment that conforms to the XML schema of the **tDeliveryError** complex type.

2.2.5 Simple Types

The following table summarizes the set of common XML schema simple type definitions defined by this specification. XML schema simple type definitions that are specific to a particular operation are described with the operation.

Simple type	Description	
tRequiredServiceTypeEnum	The delivery service required to delivery this mobile message.	

2.2.5.1 tRequiredServiceTypeEnum

The delivery service required to delivery this mobile message.

SMS_SENDER: Indicates that this message is to be delivered as a text message.

MMS_SENDER: Indicates that this message is to be delivered as a multimedia message.

2.2.6 Attributes

The following table summarizes the set of common XML schema attribute definitions defined by this specification. XML schema attribute definitions that are specific to a particular operation are described with the operation.

Attribute	Description
client	This attribute keeps a string of client information.

2.2.6.1 client

This attribute keeps a string of client information.

2.2.7 Groups

This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

3 Protocol Details

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL might specify differences that reflect actual Microsoft product behavior. For example, the schema definition might allow for an element to be **empty**, **null**, or **not present** but the behavior of the protocol as specified restricts the same elements to being **non-empty**, **not null**, and **present**.

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC2616]] section 10.

This protocol allows protocol servers to notify protocol clients of application-level faults using SOAP faults. Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

This protocol allows protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults either using HTTP status codes or using SOAP faults as specified previously in this section.

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The protocol server maintains the following states that allow a client to understand how this server accepts or handles the data (which will be text message or multimedia message delivery):

- Support of delivery of text message, multimedia message, or both.
- Limitation of text message or multimedia message supported by this server, such as maximum number of recipients.
- Support of concatenated text message or not.
- Support for batch mode delivery or not.

The protocol server also maintains a database of users and only the client of the authenticated users can send in XML containing text message or multimedia message to server.

In two-way communication application, the server MAY collect the mobile message replied from recipient's phone and send it to a client accordingly.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of WSDL operations as defined by this specification:

Operation	Description
GetServiceInfo	Specifies the properties of this Web service. Returns a GetServiceInfoResponse .
GetUserInfo	Specifies the user's information. Returns a GetUserInfoResponse .
DeliverXms	Sends one mobile message to this Web service. Returns a DeliverXmsResponse . The protocol server SHOULD<1> implement a SendXms operation that is the same as the DeliverXms operation.
DeliverXmsBatch	Sends multiple mobile messages in a batch to this Web service. Returns a DeliverXmsBatchResponse .<2>

The server also sends reply message to the client after collecting them from the recipient's phone. The server processing rules are specified in section 3.1.4.5.

3.1.4.1 GetServiceInfo

Specifies the properties of this Web service.

```
<wsdl:operation name="GetServiceInfo">
  <wsdl:input message="tns:GetServiceInfoSoapIn" />
  <wsdl:output message="tns:GetServiceInfoSoapOut" />
  </wsdl:operation>
```

The client sends a **GetServiceInfoSoapIn** request message and the server responds with a **GetServiceInfoSoapOut** response message.

The **GetServiceInfoSoapOut** response message contains basic properties of this Web service, such as the information specified in section <u>3.1.4.1.2.3</u>.

3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
GetServiceInfoSoapIn	The request to specify the properties of this Web service.
GetServiceInfoSoapOut	The response to the request to specify the properties of this Web service.

3.1.4.1.1.1 GetServiceInfoSoapIn

The message represents the client request for the server information from the server.

The **SOAP action** value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/GetServiceInfo

The **SOAP body** contains a **GetServiceInfo** element.

3.1.4.1.1.2 GetServiceInfoSoapOut

The message represents the protocol server response for a client request for the server information.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/office/Outlook/2006/OMS/GetServiceInfo
```

The SOAP body contains a **GetServiceInfoResponse** element.

3.1.4.1.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
GetServiceInfo	The request to specify the properties of this Web service.
GetServiceInfoResponse	Contains the response to the request to specify the properties of this Web service.
serviceInfo	Information for a protocol server.

3.1.4.1.2.1 GetServiceInfo

GetServiceInfo is an empty structure indicating a **GetServiceInfo** WSDL operation is being requested.

```
<xs:element name="GetServiceInfo">
  <xs:complexType />
</xs:element>
```

3.1.4.1.2.2 GetServiceInfoResponse

This structure contains the response to a **GetServiceInfo** WSDL operation.

GetServiceInfoResult: Information for a protocol server; MUST be an XML fragment that conforms to the XML schema of the **serviceInfo** element.

3.1.4.1.2.3 serviceInfo

Information for a protocol server; MUST be an XML fragment that conforms to the XML schema of the **tServiceInfo** complex type.

3.1.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
tServiceInfo	Information for a protocol server.
tSupportedService	The type of message delivery service this server supports.
tSMS_SENDER	The limitations or properties of text message delivery supported by this server.
tLONG_SMS_SENDER	The limitations or properties of concatenated text message delivery supported by this server.
tMMS_SENDER	The limitations or properties of multimedia message delivery supported by this server.

3.1.4.1.3.1 tServiceInfo

Information for a protocol server.

serviceProvider: Company name of the service provider hosting this Web service.

serviceUri: The Uniform Resource Identifier (URI) of this Web service.

signUpPage: URI of sign-up or logon page for the users of this Web service.

targetLocale: Language code identifier (LCID), as described in [MS-LCID]. Used to indicate the country or region for which this Web service is targeted. Default value is zero ("0"), which indicates that this Web service supports users globally.

localName: The name of this Web service in the language preferred by the service provider.

englishName: The name of this Web service in English.

authenticationType: Indicates the method of **authentication** supported by this Web service. MUST conform to the XML schema of the **tAuthenticationTypeEnum** simple type specified in section 3.1.4.1.4.1.

batchSize: Indicates the maximum number of mobile messages to be delivered in a single XML. The default value of zero ("0") means that the server cannot deliver multiple messages in a single XML.

supportedService: The supported message delivery service by this server; MUST conform to the XML schema of the **tSupportedService** complex type specified in section <u>3.1.4.1.3.2</u>.

3.1.4.1.3.2 tSupportedService

The **tSupportedService** complex type contains the type of message delivery service this server supports.

SMS_SENDER: Indicates that this server supports text message delivery. It MUST conform to the XML schema of the **tSMS_SENDER** complex type specified in section 3.1.4.1.3.3.

MMS_SENDER: Indicates that this server supports multimedia message delivery. It MUST conform to the XML schema of the **tMMS_SENDER** complex type specified in section 3.1.4.1.3.5.

Both elements here are optional, but at least one of them MUST be supported.

3.1.4.1.3.3 tSMS_SENDER

The **tSMS_SENDER** complex type specifies the limitations or properties of text message delivery supported by this server.

LONG_SMS_SENDER: Indicates that this server supports delivery of concatenated text message. It MUST conform to the XML schema of the **tLONG_SMS_SENDER** complex type specified in section 3.1.4.1.3.4.

maxRecipientsPerMessage: Specifies the maximum number of recipients allowed for delivering a text message.

maxMessagesPerSend: Specifies the maximum number of separate text messages allowed in a single **xmsData** string. **xmsData** is specified in section <u>3.1.4.3.2.3</u>.

maxSbcsPerMessage: Specifies the maximum number of characters allowed for a text message purely consisting of US ASCII characters.

maxDbcsPerMessage: Specifies the maximum number of characters allowed for a text message containing double byte characters.

3.1.4.1.3.4 tLONG_SMS_SENDER

The **tLONG_SMS_SENDER** complex type specifies the limitations or properties of concatenated text message delivery supported by this server.

maxRecipientsPerMessage: Specifies the maximum number of recipients allowed for delivering a concatenated text message.

maxMessagesPerSend: Specifies the maximum number of separate concatenated text messages allowed in a single **xmsData** string. **xmsData** is specified in section <u>3.1.4.3.2.3</u>.

maxSbcsPerMessage: Specifies the maximum number of characters allowed for a text message inside a concatenated text message purely consisting of US ASCII characters.

maxDbcsPerMessage: Specifies the maximum number of characters allowed for a text message inside a concatenated text message containing double byte characters.

3.1.4.1.3.5 **tMMS_SENDER**

The **tMMS_SENDER** complex type specifies the limitations or properties of multimedia message delivery supported by this server.

supportSlide: Indicates whether Synchronized Multimedia Integration Language (SMIL) is supported in describing presentation of the multimedia message or not.

maxRecipientsPerMessage: Specifies the maximum number of recipients allowed for delivering a multimedia message.

maxSizePerMessage: Specifies the maximum size, in bytes, of the multimedia message.

maxSlidesPerMessage: Specifies the maximum number of slides a multimedia message can have.

3.1.4.1.4 Simple Types

The following table summarizes the XML schema simple type definitions that are specific to this operation.

Simple type	Description
tAuthenticationTypeEnum	The method of authentication supported by this Web service.

3.1.4.1.4.1 tAuthenticationTypeEnum

Indicates the method of authentication supported by this Web service.

MUST be one of the values described in the following table.

Value	Meaning
	Users will be authenticated by passport authentication method. Passport ID and password will be used.
	Users will be authenticated by certificate over Secure Sockets Layer. No password is required.
other	Users will be authenticated by user name and password.

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.4.2 GetUserInfo

Used to retrieve the information of an authenticated user.

```
<wsdl:operation name="GetUserInfo">
  <wsdl:input message="tns:GetUserInfoSoapIn" />
```

```
<wsdl:output message="tns:GetUserInfoSoapOut" />
</wsdl:operation>
```

The client sends a **GetUserInfoSoapIn** request message, which contains the authentication information for a user, and the server responds with a **GetUserInfoSoapOut** response message, which contains the information of this authenticated user.

3.1.4.2.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
GetUserInfoSoapIn	The request to retrieve the information of an authenticated user.
GetUserInfoSoapOut	The response to a request to retrieve the information of an authenticated user.

3.1.4.2.1.1 GetUserInfoSoapIn

The message represents the client request for the user information from the server.

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/GetUserInfo

The SOAP body contains a **GetUserInfo** element.

3.1.4.2.1.2 GetUserInfoSoapOut

The message represents the protocol server response for a client request for the user information.

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/GetUserInfo

The SOAP body contains a **GetUserInfoResponse** element.

3.1.4.2.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
GetUserInfo	The request to retrieve the information of an authenticated user.
GetUserInfoResponse	Contains the response to a request to retrieve the information of an authenticated user.
xmsUser	Authentication information for a user.

Element	Description
userInfo	Information for an authenticated user.

3.1.4.2.2.1 GetUserInfo

The input data for a **GetUserInfo** WSDL operation.

xmsUser: Authentication information for a user; MUST be an XML fragment that conforms to the XML schema of the **xmsUser** element.

3.1.4.2.2.2 GetUserInfoResponse

This structure contains the response to a **GetUserInfo** WSDL operation.

GetUserInfoResult: User information for an authenticated user; MUST be an XML fragment that conforms to the XML schema of the **userInfo** element.

3.1.4.2.2.3 xmsUser

Authentication information for a user; MUST be an XML fragment that conforms to the XML schema of the **tXmsUser** complex type.

```
<xs:element name="xmsUser" type="tns:tXmsUser" />
```

3.1.4.2.2.4 userInfo

Information for an authenticated user; MUST be an XML fragment that conforms to the XML schema of the **tUserInfo** complex type.

```
<xs:element name="userInfo" type="tns:tUserInfo" />
```

3.1.4.2.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
tXmsUser	The password and user identification of a user.
tUserInfo	The mobile number and SMTP address of a user.
tUserError	Error data returned by the protocol server in response to a request.

3.1.4.2.3.1 tXmsUser

User identification and password of the user.

userId: The identification of a user. **password:** The password of a user.

customData: The protocol server MUST ignore this value.

client: See section 2.2.6.1.

3.1.4.2.3.2 tUserInfo

Information for an authenticated user.

replyPhone: The mobile number that the user used to sign up for the service with the service provider.

smtpAddress: A unique SMTP address that is used by the protocol server to deliver the reply from phone to the protocol client.

error: Error data returned by the protocol server in response to a request; MUST be an XML fragment that conforms to the XML schema of the **tUserError** complex type.

customData: Protocol server MUST ignore this value.

3.1.4.2.3.3 tUserError

Error data returned by the protocol server in response to a request.

code: If the call to **GetUserInfo** was successful, the code attribute MUST be set to "OK". If the call failed, the error code MUST be set follow the different situations in the following table or define its own protocol.

Value	Meaning
invalidUser	Invalid or unrecognized password.
unregisteredUser	User has not registered for the service.
	User's prepayment is used up or the registration is expired. The error code is for the service provider who provides prepaid service.

severity: When this attribute is returned, if the call to **GetUserInfo** was successful, the value MUST be "neutral"; otherwise it MUST be "failure".

3.1.4.2.4 Simple Types

None.

3.1.4.2.5 Attributes

None.

3.1.4.2.6 Groups

None.

3.1.4.2.7 Attribute Groups

None.

3.1.4.3 DeliverXms

Used to send a mobile message to the protocol server.

```
<wsdl:operation name="DeliverXms">
  <wsdl:input message="tns:DeliverXmsSoapIn" />
  <wsdl:output message="tns:DeliverXmsSoapOut" />
  </wsdl:operation>
```

The client sends a **DeliverXmsSoapIn** request message, which contains the content of a mobile message, and the server responds with a **DeliverXmsSoapOut** response message, which contains one or more error elements that indicate the success or failure of the attempt to send this message to the intended recipient.

3.1.4.3.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
DeliverXmsSoapIn	The request to send a mobile message to the protocol server.
DeliverXmsSoapOut	The response to a request to send a mobile message to the protocol server.

3.1.4.3.1.1 DeliverXmsSoapIn

The message represents the client request to send a mobile message to the server.

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXms

The SOAP body contains a **DeliverXms** element.

3.1.4.3.1.2 DeliverXmsSoapOut

The message represents the protocol server response that indicates the success or failure of the attempt to send this message to the intended recipient.

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXms

The SOAP body contains a **DeliverXmsResponse** element.

3.1.4.3.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
DeliverXms	The request to send a mobile message to the protocol server.
DeliverXmsResponse	Contains the response to a request to send a mobile message to the protocol server.
xmsData	The content of a mobile message.
xmsResponse	Indicates the success or failure of the attempt to send this message to the intended recipient.

3.1.4.3.2.1 DeliverXms

The input data for a **DeliverXms** WSDL operation.

xmsData: Content of a mobile message; MUST be an XML fragment that conforms to the XML schema of the **xmsData** element specified at section <u>3.1.4.3.2.3</u>.

3.1.4.3.2.2 DeliverXmsResponse

This structure contains the response to a **DeliverXms** WSDL operation.

DeliverXmsResult: One or more error elements that indicate the success or failure of the attempt to send the message to intended recipient; MUST be an XML fragment that conforms to the XML schema of the **xmsResponse** element specified at section <u>3.1.4.3.2.4</u>.

3.1.4.3.2.3 xmsData

Content of mobile message; MUST be an XML fragment that conforms to the XML schema of the **tXmsData** complex type.

```
<xs:element name="xmsData" type="tns:tXmsData" />
```

3.1.4.3.2.4 xmsResponse

Indicates the success or failure of the attempt to send this message to the intended recipient.

```
<xs:element name="xmsResponse" type="tns:tXmsResponse" />
```

It MUST be an XML fragment that conforms to the XML schema of the **tXmsResponse** complex type.

3.1.4.3.3 Complex Types

None.

3.1.4.3.4 Simple Types

None.

3.1.4.3.5 Attributes

3.1.4.3.6 Groups

None.

3.1.4.3.7 Attribute Groups

None.

3.1.4.4 DeliverXmsBatch

Used to send a batch of mobile messages to the protocol server.

```
<wsdl:operation name="DeliverXmsBatch">
  <wsdl:input message="tns:DeliverXmsBatchSoapIn" />
  <wsdl:output message="tns:DeliverXmsBatchSoapOut" />
  </wsdl:operation>
```

The client sends a **DeliverXmsBatchSoapIn** request message, which contains the content of a batch of mobile messages, and the server responds with a **DeliverXmsBatchSoapOut** response message, which contains one or more error elements that indicate the success or failure of the attempt to send each and every of these messages to the intended recipient.

3.1.4.4.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
DeliverXmsBatchSoapIn	The request to send a batch of mobile messages to the protocol server.
DeliverXmsBatchSoapOut	The response to a request to send a batch of mobile messages to the protocol server.

3.1.4.4.1.1 DeliverXmsBatchSoapIn

The message represents the client request to send a batch of mobile messages to the server.

The SOAP action value of the message is defined as:

 $\verb|http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXmsBatch| \\$

The SOAP body contains a **DeliverXmsBatch** element.

3.1.4.4.1.2 DeliverXmsBatchSoapOut

The message represents the protocol server response that indicates the success or failure of the attempt to each and every of the messages to the intended recipient.

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXmsBatch

The SOAP body contains a **DeliverXmsBatchResponse** element.

3.1.4.4.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
DeliverXmsBatch	The request to send a batch of mobile messages to the protocol server.
DeliverXmsBatchResponse	Contains the response to a request to send a batch of mobile messages to the protocol server.
xmsBatch	The content of a batch of mobile messages.
xmsResponses	Indicates the success or failure of the attempt to send this message to the intended recipients.

3.1.4.4.2.1 DeliverXmsBatch

The input data for a **DeliverXmsBatch** WSDL operation.

packageXml: Content of a batch of mobile messages; MUST be an XML fragment that conforms to the XML schema of the **xmsBatch** element.

3.1.4.4.2.2 DeliverXmsBatchResponse

This structure contains the response to a **DeliverXmsBatch** WSDL operation.

DeliverXmsBatchResult: One or more error elements that indicate the success or failure of the attempt to send each and every of a batch of the messages to intended recipient; MUST be an XML fragment that conforms to the XML schema of the **xmsResponses** element.

3.1.4.4.2.3 xmsBatch

```
<xs:element name="xmsBatch" type="tns:tXmsBatch" />
```

xmsBatch: Content of a batch of mobile messages; MUST be an XML fragment that conforms to the XML schema of the **tXmsBatch** complex type.

3.1.4.4.2.4 xmsResponses

xmsResponse: Indicates the success or failure of the attempt to send this message to the intended recipient; MUST be an XML fragment that conforms to the XML schema of the **tXmsResponseWithId** complex type.

3.1.4.4.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
tXmsBatch	Content of mobile message.
tXmsDataInBatch	The content of a mobile message, using the id attribute to identify each and every message in a specific batch.
tXmaResponseWithId	Indicates the success or failure of the attempt to send this message to the intended recipients, using the id attribute for identification of the message.

3.1.4.4.3.1 tXmsBatch

xmsData: Content of mobile message; MUST be an XML fragment that conforms to the XML schema of the **tXmsDataInBatch** complex type.

client: See section 2.2.6.1.

3.1.4.4.3.2 tXmsDataInBatch

The **tXmsDataInBatch** is similar to the **xmsData** of the **DeliverXms** operation, except that it uses the **id** attribute to identify each and every message in a specific batch.

```
</xs:sequence>
  <xs:attribute name="Id" type="xs:unsignedInt" use="required" />
</xs:complexType>
```

user: See section <u>2.2.4.17</u>.

xmsHead: See section 2.2.4.17.
xmsBody: See section 2.2.4.17.

Id: Identifies each and every message in a specific batch.

3.1.4.4.3.3 tXmsResponseWithId

The **tXmsResponseWithId** is similar to the **xmsResponse** of the **DeliverXms** operation, except that it uses attribute **id** to identify each and every message in a specific batch.

error: See section 2.2.4.19.

id: Identifies each and every message in a specific batch.

3.1.4.4.4 Simple Types

None.

3.1.4.4.5 Attributes

None.

3.1.4.4.6 Groups

None.

3.1.4.4.7 Attribute Groups

None.

3.1.4.5 Send reply message to client after collecting them from the recipient's phone

To enable two-way mobile message communication between the client and a mobile phone, the protocol server is required to package the reply sent from a mobile phone into a MIME-formatted SMTP e-mail message specified in the section 2.2.2.1, and then send the e-mail messages to a user-designated e-mail address.

3.1.5 Timer Events

3.1.6 Other Local Events

None.

3.2 Client Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

When there is reply to the mobile message from the recipient's phone, the protocol server sends the reply to client via the SMTP protocol. If the application requires two-way communication from client to server and needs the client to receive reply messages, the client MAY implement the details specified in this section.

3.2.2 Timers

None.

3.2.3 Initialization

To accept the reply of the mobile message from the protocol server, the client MUST be able to retrieve e-mail messages from an e-mail address that the server will send the reply message to.

3.2.4 Message Processing Events and Sequencing Rules

When a client receives the SMTP e-mail messages, the client MUST understand the message as specified in section 2.2.2.1. The client MUST recognize the content class and treat it as a mobile message. The client MUST be able to recognize them as text or multimedia messages. When these messages are opened, replied to, or forwarded, they are automatically treated as mobile messages. That also means, in the message reply and forward operations, the client MUST be able to allow its user to send the mobile message via calling the server's **DeliverXms** WSDL operation as specified in section 3.1.4.3.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

4 Protocol Examples

The following provides examples of the interaction between protocol client and protocol server.

4.1 GetServiceInfo

An OMS client can submit a **GetServiceInfo** request with an empty parameter with the following code:

The protocol server receives this request and creates an appropriate response, similar to the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<serviceInfo>
  <serviceProvider>ABC Company</serviceProvider>
  <serviceUri>http://www.abc.com.cn/OMS3/XMS.asmx</serviceUri>
 <signUpPage>http://www.abc.com.cn/ws/xmssignup.aspx/</signUpPage>
  <targetLocale>2052</targetLocale>
  <localName>ABC Mobile Service</localName>
  <englishName>ABC Mobile Service</englishName>
  <authenticationType>other</authenticationType>
  <batchSize>255</batchSize>
  <supportedService>
      <SMS SENDER maxRecipientsPerMessage="50"
                  maxMessagesPerSend="20"
                  maxSbcsPerMessage="140"
                  maxDbcsPerMessage="70" >
         <LONG SMS SENDER maxRecipientsPerMessage="50"</pre>
                  maxMessagesPerSend="255"
                  maxSbcsPerMessage="153"
                  maxDbcsPerMessage="67" />
      </SMS SENDER>
      <MMS SENDER supportSlide="true"</pre>
                  maxRecipientsPerMessage="100"
                  maxSizePerMessage="30000"
                  maxSlidesPerMessage="10" />
   </supportedService>
</serviceInfo>
```

4.2 GetUserInfo

When the client wants to retrieve user information of an authenticated user in the protocol server, the request would resemble the following code:

The following code is returned from the protocol server after a successful call:

```
<?xml version="1.0" encoding="utf-8"?>
<userInfo xmlns = "http://schemas.microsoft.com/office/Outlook/2006/OMS">
        <replyPhone>090123456</replyPhone>
        <smtpAddress>userid.spmail@spdomain.com</smtpAddress>
        <error code="ok"/>
        </userInfo>
```

The following code is returned from the protocol server after an error:

4.3 DeliverXms

When the client wants to send a text message to the protocol server, the request would resemble the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<xmsData client="Microsoft Office Outlook 12.0"</pre>
        xmlns = "http://schemas.microsoft.com/office/Outlook/2006/OMS">
  <user>
    <userId>myname</userId>
    <password>mypwd</password>
    <replyPhone>13801391350</replyPhone>
    <customData/>
  </11ser>
  <msHead>
    <scheduled>2005-04-20T14:20:00Z</scheduled>
    <requiredService>SMS SENDER</requiredService>
      <recipient>135xxxx</recipient>
      <recipient>139xxxx</recipient>
    </t.o>
  </xmsHead>
  <xmsBody format="SMS">
    <content contentType="text/plain"</pre>
      contentId="Att0.txt@AB1B43B2B0594564.B94EF7ABB12B49BA"
      contentLocation="1.txt">(1/2)This is the first SMS message...</content>
    <content contentType="text/plain"</pre>
      contentId="Att1.txt@AB1B43B2B0594564.B94EF7ABB12B49BA"
      contentLocation="2.txt">(2/2)This is the second SMS message...</content>
  </xmsBody>
</xmsData>
```

When the client wants to send a multimedia message to the protocol server, the request would resemble the following code:

```
<scheduled>2005-04-20T14:20:00Z</scheduled>
    <reguiredService>MMS SENDER</reguiredService>
    <sourceType>reminder</sourceType>
    <to>
      <recipient>135xxxx</recipient>
      <recipient>139xxxx</recipient>
    </to>
    <subject>My Message</subject>
  </xmsHead>
  <xmsBody format="MMS">
    <mmsSlides>
      <head>
        <meta name="author" content="msOfficeOutlookOms" />
        <lavout>
          <root-layout width="120" height="120" background-color="#ffffff" />
          <region id="Image" left="0" top="0" width="120" height="90" />
          <region id="Text" left="0" top="90" width="120" height="30" />
        </layout>
      </head>
      <body>
        <par dur="3000">
          <imq src="cid:Att1.qif@AB1B43B2B0594564.B94EF7ABB12B49BA"</pre>
              region="Image" />
          <text src="cid:Att0.txt@AB1B43B2B0594564.B94EF7ABB12B49BA"</pre>
               region="Text"/>
          <audio src="cid:Att2.mid@AB1B43B2B0594564.B94EF7ABB12B49BA" />
      </par>
      </body>
    </mmsSlides>
    <content contentType="text/plain"</pre>
             contentId="Att0.txt@AB1B43B2B0594564.B94EF7ABB12B49BA"
             contentLocation="1.txt">This is the text part</content>
    <content contentType="image/gif"</pre>
             contentId="Att1.gif@AB1B43B2B0594564.B94EF7ABB12B49BA"
             contentLocation=
                  "106675.gif">/9j/4AAQ ..... AVExISEyccHhcgLikxMC4p</content>
    <content contentType="audio/mid"</pre>
             contentId="Att2.mid@AB1B43B2B0594564.B94EF7ABB12B49BA"
             contentLocation="1898.mid">/wDQjVYUrl ..... GoJ4e8j</content>
 </xmsBodv>
</xmsData>
```

After receiving the previous request, the following response is an example generated by protocol server when the message is delivered successfully:

The following response is an example generated by protocol server when the message failed to be delivered:

4.4 DeliverXmsBatch

The following code is an example from client for sending a batch of mobile messages (two text messages in this example):

```
<?xml version="1.0" encoding="utf-16"?>
<xmsBatch client="Microsoft Windows SharePoint Service"</pre>
         xmlns="http://schemas.microsoft.com/office/Outlook/2006/OMS">
  <xmsData id="0">
    <user>
      <userId>ddguo</userId>
      <password />
      <customData />
    </user>
    <xmsHead>
      <requiredService>SMS SENDER</requiredService>
      <sourceType>wssAlert
        <recipient>13671121236</recipient>
      </to>
    </xmsHead>
    <xmsBody format="SMS">
      <content contentType="text/plain"</pre>
               contentId="1.txt@5ca13ed023024ed59cfae6c0e185a5db"
               contentLocation="1.txt">This is a testing message.</content>
   </xmsBody>
  </xmsData>
  <xmsData id="1">
   <user>
      <userId>ddguo</userId>
      <password />
      <customData />
    </user>
    <xmsHead>
      <requiredService>SMS SENDER</requiredService>
      <sourceType>wssAlert
        <recipient>13671121236</recipient>
      </to>
    </xmsHead>
    <xmsBody format="SMS">
      <content contentType="text/plain"</pre>
               contentId="1.txt@ecf25304326e497c8775a929a3178311"
               contentLocation="1.txt">This is a testing message.</content>
    </msBody>
  </xmsData>
</xmsBatch>
```

The following code is an example of response from the protocol server for the able request:

4.5 Send Reply Message from Server to Client after Server Collects the Mobile Message from Recipient's Phone

The following is an example of an incoming text message that is packaged as e-mail message:

```
From: "Mobile Inbound Agent" incomingmessage@service-provider.com
To: someone@example.com
Subject: This is a text message
Date: Mon, 7 Nov 2005 17:52:00 +0800
Content-class: MS-OMS-SMS
X-MS-Reply-to-mobile: +8613601391354
MIME-Version: 1.0
Content-Type: text/plain; charset="gb2312"
Content-Transfer-Encoding: quoted-printable
This is a text message from a mobile phone replying to a text message from Outlook.
```

The following is an example of an incoming multimedia message that is packaged as e-mail message:

```
From: "Mobile Inbound Agent" incomingmessage@service-provider.com
To: someone@example.com
Subject: This is a multimedia message (Open the message to view its content)
Date: Mon, 7 Nov 2005 17:52:00 +0800
Content-class: MS-OMS-MMS
X-MS-Reply-to-mobile: +8613601391354
MIME-Version: 1.0
Content-Type: multipart/related; type="application/smil";
boundary="-----Boundary=_thisisboundary"
This is a multi-part message in MIME format.
-----Boundary=_thisisboundary
Content-Type: application/smil; name="mmspresent.smil"
Content-Location: "mmspresent.smil"
Content-Transfer-Encoding: Base64
PHNtaWw+... 1pbD4=
-----Boundary= thisisboundary
Content-Type: text/plain; name="textpart.txt"
Content-Transfer-Encoding: Base64
Content-Location: textpart.txt
6Zi/5YWs5Y+45rOV5b6L5biI6IyD5Zu057uV6YGT6LCi
-----Boundary=_thisisboundary
Content-Type: image/gif; name="imagepart.gif"
Content-Transfer-Encoding: Base64
Content-Location:imagepart.gif
R01GODlheABaAPf/...BDQi6j4uQAxwcixRzZErI5ROjfvSHJcmRMGBAAOw==
-----Boundary=_thisisboundary
Content-Type: audio/mid; name="audiopart.mid"
Content-Transfer-Encoding: Base64
Content-Location: audiopart.mid
TVRoZAAAAAY...XBDfwA/fwA6f4dAOqAAPwAAQwAA/y8A
    -----Boundary=_thisisboundary
```

5 Security

5.1 Security Considerations for Implementers

This protocol introduces no additional security considerations beyond those applicable to its underlying protocols.

5.2 Index of Security Parameters

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL and schema are provided in this appendix.

```
<?xml version="1.0"?>
<wsdl:definitions</pre>
 targetNamespace="http://schemas.microsoft.com/office/Outlook/2006/OMS"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
 xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
 xmlns:tns="http://schemas.microsoft.com/office/Outlook/2006/OMS"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
 xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" >
 <wsdl:tvpes>
    <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"</pre>
        targetNamespace="http://schemas.microsoft.com/office/Outlook/2006/OMS">
      <!-- GetServiceInfo: The Schema of Response Xml-->
      <xs:simpleType name="tAuthenticationTypeEnum">
        <xs:restriction base="xs:string">
          <xs:enumeration value="passport" />
          <xs:enumeration value="fulltrust" />
          <xs:enumeration value="other" />
        </xs:restriction>
      </xs:simpleType>
      <xs:complexType name="tLONG SMS SENDER">
        <xs:attribute name="maxRecipientsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxMessagesPerSend" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxSbcsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxDbcsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
      </xs:complexType>
      <xs:complexType name="tSMS SENDER">
        <xs:sequence>
          <xs:element name="LONG SMS SENDER" type="tns:tLONG SMS SENDER"</pre>
                      minOccurs="0" />
        </xs:sequence>
        <xs:attribute name="maxRecipientsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxMessagesPerSend" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxSbcsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxDbcsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
      </xs:complexType>
      <xs:complexType name="tMMS SENDER">
        <xs:attribute name="supportSlide" type="xs:boolean" use="required" />
        <xs:attribute name="maxRecipientsPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxSizePerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
        <xs:attribute name="maxSlidesPerMessage" type="xs:unsignedInt"</pre>
                      use="required" />
      </xs:complexType>
      <xs:complexType name="tSupportedService">
        <xs:sequence minOccurs="1" maxOccurs="2">
          <xs:element name="SMS SENDER" type="tns:tSMS SENDER" minOccurs="0" />
          <xs:element name="MMS SENDER" type="tns:tMMS SENDER" minOccurs="0" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tServiceInfo">
```

```
<xs:element name="serviceProvider" type="xs:string" />
<xs:element name="serviceUri" type="xs:string" />
    <xs:element name="signUpPage" type="xs:string" />
    <xs:element name="targetLocale" type="xs:unsignedShort"</pre>
                minOccurs="0" default="0" />
    <xs:element name="localName" type="xs:string" />
    <xs:element name="englishName" type="xs:string" />
    <xs:element name="authenticationType"</pre>
                type="tns:tAuthenticationTypeEnum" />
    <xs:element name="batchSize" type="xs:unsignedInt"</pre>
                minOccurs="0" default="0" />
    <xs:element name="supportedService" type="tns:tSupportedService" />
 </xs:sequence>
</xs:complexType>
<xs:element name="GetServiceInfo">
  <xs:complexType />
</xs:element>
<xs:element name="serviceInfo" type="tns:tServiceInfo" />
<xs:element name="GetServiceInfoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="GetServiceInfoResult" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- GetUserInfo Method: The Schema of xmsUser Xml -->
<xs:complexType name="tXmsUser">
  <xs:sequence>
    <xs:element name="userId" type="xs:string" minOccurs="0" />
    <xs:element name="password" type="xs:string" minOccurs="0" />
    <xs:element name="customData" type="xs:string" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="client" type="xs:string" />
</xs:complexType>
<xs:element name="xmsUser" type="tns:tXmsUser" />
<xs:element name="GetUserInfo">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="xmsUser" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- GetUserInfo Method: The Schema of Response Xml -->
<xs:complexType name="tUserError">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="code" type="xs:string" use="required" />
      <xs:attribute name="severity" type="xs:string" use="optional" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="tUserInfo">
  <xs:sequence>
    <xs:element name="replyPhone" type="xs:string" minOccurs="0" />
    <xs:element name="smtpAddress" type="xs:string" minOccurs="0" />
    <xs:element name="error" type="tns:tUserError" />
    <xs:element name="customData" type="xs:string" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
<xs:element name="userInfo" type="tns:tUserInfo" />
<xs:element name="GetUserInfoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="GetUserInfoResult" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
```

```
</xs:element>
      <!-- DeliverXms Method: The Schema of xmsData Xml -->
      <xs:simpleType name="tRequiredServiceTypeEnum">
          <xs:restriction base="xs:string">
              <xs:enumeration value="SMS SENDER" />
              <xs:enumeration value="MMS SENDER" />
          </xs:restriction>
      </xs:simpleType>
      <xs:complexType name="tTo">
        <xs:sequence>
          <xs:element maxOccurs="unbounded" name="recipient" type="xs:string" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tXmsHeader">
        <xs:sequence>
          <xs:element name="scheduled" type="xs:dateTime" minOccurs="0" />
          <xs:element name="requiredService" type="tns:tRequiredServiceTypeEnum"</pre>
minOccurs="0" />
          <xs:element name="sourceType" type="xs:string" minOccurs="0" />
          <xs:element name="to" type="tns:tTo" />
          <xs:element name="subject" type="xs:string" minOccurs="0" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tUser">
          <xs:element name="userId" type="xs:string" minOccurs="0" />
<xs:element name="password" type="xs:string" minOccurs="0" />
          <xs:element name="replyPhone" type="xs:string" minOccurs="0" />
          <xs:element name="customData" type="xs:string" minOccurs="0" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tMeta">
        <xs:attribute name="name" type="xs:string" use="required" />
<xs:attribute name="content" type="xs:string" use="required" />
      </xs:complexType>
      <xs:complexType name="tRoot-layout">
        <xs:attribute name="width" type="xs:unsignedInt" use="required" />
        <xs:attribute name="height" type="xs:unsignedByte" use="required" />
        <xs:attribute name="background-color" type="xs:string" use="required" />
      </xs:complexTvpe>
      <xs:complexType name="tRegion">
        <xs:attribute name="id" type="xs:string" use="required" />
        <xs:attribute name="left" type="xs:string" use="required" />
        <xs:attribute name="top" type="xs:string" use="required" />
        <xs:attribute name="width" type="xs:string" use="required" />
        <xs:attribute name="height" type="xs:string" use="required" />
        <xs:attribute name="fit" type="xs:string" use="required" />
      </xs:complexType>
      <xs:complexType name="tLayout">
        <xs:sequence>
          <xs:element name="root-layout" type="tns:tRoot-layout" />
          <xs:element name="region" type="tns:tRegion" maxOccurs="2" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tHeader">
        <xs:sequence>
          <xs:element name="meta" type="tns:tMeta" minOccurs="0" />
          <xs:element name="layout" type="tns:tLayout" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="tImg">
        <xs:attribute name="src" type="xs:string" use="required" />
        <xs:attribute name="region" type="xs:string" use="required" />
      </xs:complexType>
      <xs:complexType name="tText">
        <xs:attribute name="src" type="xs:string" use="required" />
        <xs:attribute name="region" type="xs:string" use="required" />
        <xs:attribute name="foreground-color" type="xs:string" use="optional" />
```

```
</xs:complexType>
<xs:complexType name="tAudio">
  <xs:attribute name="src" type="xs:string" use="required" />
</xs:complexType>
<xs:complexType name="tPar">
  <xs:sequence>
   <xs:element name="img" type="tns:tImg" minOccurs="0" />
    <xs:element name="text" type="tns:tText" minOccurs="0" />
    <xs:element name="audio" type="tns:tAudio" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="dur" type="xs:string" use="required" />
</xs:complexType>
<xs:complexType name="tBody">
  <xs:sequence>
    <xs:element name="par" type="tns:tPar" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>
<xs:complexType name="tMmsSlides">
  <xs:sequence>
    <xs:element name="head" type="tns:tHeader" minOccurs="0" />
    <xs:element name="body" type="tns:tBody" />
  </xs:sequence>
</xs:complexType>
<xs:complexType name="tContent">
 <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="contentType" type="xs:string" use="required" />
      <xs:attribute name="contentId" type="xs:string" use="required" />
      <xs:attribute name="contentLocation" type="xs:string" use="required" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="tXmsBody">
  <xs:sequence>
    <xs:element name="mmsSlides" type="tns:tMmsSlides" minOccurs="0" />
    <xs:element name="content" type="tns:tContent" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="format" type="xs:string" use="required" />
</xs:complexType>
<xs:complexType name="tXmsData">
  <xs:sequence>
   <xs:element name="user" type="tns:tUser" minOccurs="0" />
    <xs:element name="xmsHead" type="tns:tXmsHeader" minOccurs="0" />
   <xs:element name="xmsBody" type="tns:tXmsBody" />
  </xs:sequence>
  <xs:attribute name="client" type="xs:string" />
</xs:complexType>
<xs:element name="xmsData" type="tns:tXmsData" />
<xs:element name="DeliverXms">
  <xs:complexType>
   <xs:sequence>
      <xs:element name="xmsData" type="xs:string" />
   </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- DeliverXms Method: The Schema of Response Xml -->
<xs:complexType name="tDeliveryError">
  <xs:sequence>
   <xs:element name="content" type="xs:string" minOccurs="0" />
    <xs:element name="recipientList" type="xs:string" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="code" type="xs:string" use="required" />
  <xs:attribute name="severity" type="xs:string" use="required" />
</xs:complexType>
<xs:complexType name="tXmsResponse">
  <xs:sequence>
    <xs:element name="error" type="tns:tDeliveryError" maxOccurs="unbounded" />
  </xs:sequence>
```

```
</xs:complexType>
    <xs:element name="xmsResponse" type="tns:tXmsResponse" />
    <xs:element name="DeliverXmsResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="DeliverXmsResult" type="xs:string" />
        </r></r></r>
      </xs:complexType>
    </xs:element>
    <!-- DeliverXmsBatch Method: The Schema of packageXml Xml -->
    <xs:complexType name="tXmsDataInBatch">
      <xs:sequence>
        <xs:element name="user" type="tns:tUser" minOccurs="0" />
        <xs:element name="xmsHead" type="tns:tXmsHeader" minOccurs="0" />
        <xs:element name="xmsBody" type="tns:tXmsBody" />
      </xs:sequence>
      <xs:attribute name="Id" type="xs:unsignedInt" use="required" />
    </xs:complexType>
    <xs:complexType name="tXmsBatch">
      <xs:sequence>
        <xs:element name="xmsData" type="tns:tXmsDataInBatch"</pre>
                   maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="client" type="xs:string" />
    </xs:complexType>
    <xs:element name="xmsBatch" type="tns:tXmsBatch" />
    <xs:element name="DeliverXmsBatch">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="packageXml" type="xs:string" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <!-- DeliverXmsBatch Method: The Schema of Response Xml -->
    <xs:complexType name="tXmsResponseWithId">
      <xs:sequence>
        <xs:element name="error" type="tns:tDeliveryError"</pre>
                    maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="id" type="xs:unsignedInt" use="required" />
    </xs:complexType>
    <xs:element name="xmsResponses">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="xmsResponse" type="tns:tXmsResponseWithId"</pre>
                     maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="DeliverXmsBatchResponse">
      <xs:complexTvpe>
        <xs:sequence>
          <xs:element name="DeliverXmsBatchResult" type="xs:string" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:schema>
</wsdl:types>
<wsdl:message name="GetServiceInfoSoapIn">
  <wsdl:part name="parameters" element="tns:GetServiceInfo" />
</wsdl:message>
<wsdl:message name="GetServiceInfoSoapOut">
  <wsdl:part name="parameters" element="tns:GetServiceInfoResponse" />
</wsdl:message>
<wsdl:message name="GetUserInfoSoapIn">
  <wsdl:part name="parameters" element="tns:GetUserInfo" />
</wsdl:message>
```

```
<wsdl:message name="GetUserInfoSoapOut">
    <wsdl:part name="parameters" element="tns:GetUserInfoResponse" />
  </wsdl:message>
  <wsdl:message name="DeliverXmsSoapIn">
    <wsdl:part name="parameters" element="tns:DeliverXms" />
  </wsdl:message>
  <wsdl:message name="DeliverXmsSoapOut">
    <wsdl:part name="parameters" element="tns:DeliverXmsResponse" />
  </wsdl:message>
  <wsdl:message name="DeliverXmsBatchSoapIn">
    <wsdl:part name="parameters" element="tns:DeliverXmsBatch" />
  </wsdl:message>
  <wsdl:message name="DeliverXmsBatchSoapOut">
    <wsdl:part name="parameters" element="tns:DeliverXmsBatchResponse" />
  </wsdl:message>
  <wsdl:portType name="OMSServiceSoap">
    <wsdl:operation name="GetServiceInfo">
      <wsdl:input message="tns:GetServiceInfoSoapIn" />
      <wsdl:output message="tns:GetServiceInfoSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetUserInfo">
      <wsdl:input message="tns:GetUserInfoSoapIn" />
      <wsdl:output message="tns:GetUserInfoSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="DeliverXms">
      <wsdl:input message="tns:DeliverXmsSoapIn" />
      <wsdl:output message="tns:DeliverXmsSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="DeliverXmsBatch">
      <wsdl:input message="tns:DeliverXmsBatchSoapIn" />
      <wsdl:output message="tns:DeliverXmsBatchSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="OMSServiceSoap" type="tns:OMSServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetServiceInfo">
     <soap:operation</pre>
style="document" />
     <wsdl:input>
       <soap:body use="literal" />
      </wsdl:input>
     <wsdl:output>
       <soap:body use="literal" />
     </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetUserInfo">
      <soap:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/GetUserInfo"
                     style="document" />
      <wsdl:input>
       <soap:body use="literal" />
      </wsdl:input>
     <wsdl:output>
       <soap:body use="literal" />
     </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeliverXms">
     <soap:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXms"
                     style="document" />
     <wsdl:input>
       <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
       <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
```

```
<wsdl:operation name="DeliverXmsBatch">
      <soap:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXmsBatch"
                      style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="OMSServiceSoap12" type="tns:OMSServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetServiceInfo">
      <soap12:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/GetServiceInfo"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetUserInfo">
      <soap12:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/GetUserInfo"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeliverXms">
      <soap12:operation</pre>
\verb|soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXms"|
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DeliverXmsBatch">
      <soap12:operation</pre>
soapAction="http://schemas.microsoft.com/office/Outlook/2006/OMS/DeliverXmsBatch"
                        style="document" />
      <wsdl:input>
       <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Microsoft Outlook 2016

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 3.1.4: To support Microsoft Office Outlook 2007 Service Pack 1 or Outlook 2010, it is not necessary for the protocol server to implement the **SendXms** operation.

<2> Section 3.1.4: Use the **DeliverXmsBatch** operation in implementations for SharePoint Foundation 2010 but not for Office Outlook 2007 or Outlook 2010.

8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

9 Index

A	Examples
	DeliverXms 47
Abstract data model	DeliverXmsBatch 49
client 45	GetServiceInfo 46
server 28	GetUserInfo 46
Applicability 11	overview 46
Attribute groups 27	send reply message from protocol server to
Attributes 27	protocol client 50
client 27	
	F
C	
	Fields - vendor-extensible 12
Capability negotiation 12	Full WSDL 52
Change tracking 60	
Client	G
abstract data model 45	
details 45	GetServiceInfo example 46
initialization 45	GetUserInfo example 46
local events 45	Glossary 7
message processing 45	Groups 27
sequencing rules 45	
timer events 45	I
timers 45	
client attribute 27	<u>Implementer - security considerations</u> 51
<u>Common data structures</u> 27	Index of security parameters 51
Complex types 15	Informative references 9
tAudio 17	Initialization
tBody 17	client 45
tContent 17	server 28
tDeliveryError 18	Introduction 7
tHeader 21	
tImg 21	L
tLayout 21	
tMeta 21	Local events
tMmsSlides 22	client 45
<u>tPar</u> 22	server 45
tRegion 22	
tRoot-layout 23	М
tText 23	•
<u>tTo</u> 24	Message processing
tUser 24	client 45
tXmsBody 24	server 29
tXmsData 25	Messages
tXmsHeader 25	attribute groups 27
tXmsResponse 26	attributes 27
	<u>client attribute</u> 27
D	common data structures 27
	complex types 15
<u>Data communication – scenario</u> 10	elements 15
Data model - abstract	enumerated 13
<u>client</u> 45	groups 27
server 28	Mobile message packaged as MIME formatted e-
DeliverXms example 47	mail message 14
<u>DeliverXmsBatch example</u> 49	message body 15
	incoming multimedia message 15
E	incoming text message 15
	message headers 14
Events	Content-Class 14
<u>local - client</u> 45	From 14
<u>local - server</u> 45	Subject 14
<u>timer - client</u> 45	<u>To</u> 14
<u>timer - server</u> 44	X-MS-Reply-To-Mobile 14

mail message message 14 namespaces 13 simple types 26 syntax 13 tAudio complex type 17 tBody complex type 17 tContent complex type 17 tContent complex type 18 tHeader complex type 21 tImg complex type 21 tLayout complex type 21 tMeta complex type 21 tReguired Service PypeEnum simple type 26 tReguired Service PypeEnum simple type 26 tRoot-layout complex type 23 tTex complex type 24 tXmsData complex type 24 tXmsData complex type 24 tXmsData complex type 25 tXmsResponses type 25 tXmsResponse complex type 26 message message body 15 incoming multimedia message 15 incomi		
namespaces 13 simple types 26 syntax 13 fudio complex type 17 fleedy complex type 18 fleeder complex type 12 flims complex type 21 fleeder complex type 21 fleeder complex type 21 fleeder complex type 22 fleeder complex type 22 fleeder complex type 23 fleedin complex type 23 fleedin complex type 24 fleeder complex type 24 fleeder complex type 25 fleeder layout complex type 26 fleeder layout complex type 25 fleeder complex type 26 fleeder complex type 26 fleeder complex type 27 fleeder complex type 26 fleeder complex type 27 fleeder complex type 28 fleeder complex type 26 fleeder complex type 26 fleeder complex type 27 fleeder complex type 27 fleeder complex type 28 fleeder complex type 27 fleeder complex type 28 fleeder complex type 28 fleeder complex type 29 fleeder complex type 29 fleeder complex type 29 fleeder complex type 20 fleeder complex type 20 fleeder complex type 21 fleeder complex type 26 fleeder complex type 27 fleeder complex type 27 fleeder complex type 28 fleeder complex type 28 fleeder complex type 28 fleeder complex type 28 fleeder complex type 29 fleeder complex type 29 fleeder complex type 29 fleeder complex type 20 fleeder complex t	Mobile message packaged as MIME formatted e-	attribute groups 44
simple types 26 syntax 13 tAudio complex type 17 tContent complex type 17 tContent complex type 17 tContent complex type 18 ttleader complex type 21 ttling type 21 ttling complex type 21 ttling complex type 22 transport 13 Reagion complex type 22 transport 13 trext complex type 22 transport 13 trext complex type 23 trice complex type 23 trice complex type 24 txmsBear complex type 24 txmsBear complex type 25 txmsReaponse complex type 25 txmsReaponse complex type 25 txmsReaponse complex type 26 txmsReaponse complex type 26 txmsReaponse complex type 27 txmsDeap complex type 28 txmsReaponse complex type 29 txmsReaponse complex type 26 txmsReaponse complex type 27 txmsDeap complex type 28 txmsReaponse complex type 29 txmsReaponse complex type 26 txmsReaponse complex type 26 txmsReaponse complex type 27 txmsDeap complex type 28 txmsReaponse complex type 29 txmsReaponse complex type 26 txmsReaponse complex type 26 txmsReaponse complex type 27 txmsDeap complex type 28 txmsReaponse complex type 29 txmsReaponse complex type 29 txmsReaponse complex type 26 txmsReaponse complex type 26 txmsReaponse complex type 27 txmsDeap complex type 28 txmsReaponse complex type 29 txmsReaponse complex type 20 txmsReaponse complex type 20 txmsReaponse complex type 20 dealerners N N Namespaces 13 Normative references 8 Obtaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server operation of type type 30 cellular type 30 ce		
syritax 13 EAudio complex type 17 EBody complex type 18 Header complex type 21 Iffina complex type 22 Iffina complex type 22 Iffina complex type 23 Iffination complex type 25 Iffination complex type 26 Iffination complex type 26 Iffination complex type 26 Iffination complex type 26 Iffination complex type 27 Iffination com	namespaces 13	complex types
syritax 13 EAudio complex type 17 EBody complex type 18 Header complex type 21 Iffina complex type 22 Iffina complex type 22 Iffina complex type 23 Iffination complex type 25 Iffination complex type 26 Iffination complex type 26 Iffination complex type 26 Iffination complex type 26 Iffination complex type 27 Iffination com	simple types 26	tXmsBatch 43
Exms. Examplex type 17 Ecenter complex type 17 Ecenter complex type 17 Ecenter complex type 21 Etheoder type 21 Etheoder type 21 Etheoder type 21 Etheoder type 22 Etheoder type 22 Etheoder type 22 Etheoder type 22 Etheoder type 23 Etheoder type 23 Etheoder type 24 Etheoder type 24 Etheoder type 24 Etheoder type 24 Etheoder type 25 Exms. Export 13 Exms. Export 13 Exms. Export 14 Export type 24 Exms. Export type 25 Exms. Export type 25 Exms. Export type 26 Exms. Export type 27 Export type 27 Export type 28 Exms. Export type 29 Export type 29 Extra type 29	syntax 13	tXmsDataInBatch 43
iBody complex type 17 Content complex type 18 theader complex type 21 tima complex type 22 the complex type 22 the complex type 23 transport 13 tRegion complex type 23 transport 13 tRegion complex type 23 transport 13 tRegion complex type 23 transport 13 trest complex type 24 transport 15 trest complex type 24 transport 18 trest complex type 24 transport 18 trest complex type 25 transport 18 trest complex type 26 trest complex type 27 trest complex type 27 trest complex type 28 trest complex type 28 trest complex type 29 trest complex		
Content complex type 17 DeliverYmsRatch 42 DeliverYmsRatch 42 DeliverYmsRatch 42 The complex type 21 Lavout complex type 22 Lass complex type 22 Lass complex type 22 Lass complex type 22 Lass complex type 22 Required Service Type Enum simple type 26 Required Service Type 24 Last complex type 24 Last complex type 24 Last complex type 25 Last complex type 26 Last complex type 27 Last complex type 28 Last complex type 29 Last complex type 29 Last complex type 29 Last complex type 20 Last complex type 20 Last complex type 21 Last complex type 24 Last complex type 24 Last complex type 25 Last complex type 25 Last complex type 26 Last complex type 27 Last complex type 28 Last complex type 29 Last complex type 21 Last complex type 21 Last complex type 21 Last complex type 21 Last complex type 23 Last complex type 23 Last complex type 24 Last complex type 24 Last complex type 26 Last complex type 27 Last complex type 28 Last complex type 28 Last complex type 29 Last complex typ		
DeliverXmsRatchResponse 42 theader complex type 21 theader complex type 22 theader complex type 22 theader complex type 22 theader complex type 22 theader complex type 23 theader complex type 23 theader complex type 23 theader complex type 24 the special complex type 24 the special complex type 24 the special complex type 25 the special complex type 26 the special complex type 27 the special complex type 27 the special complex type 27 the special complex type 28 the special complex type 28 the special complex type 29 the special complex type 20 the special com		
titleader complex type 21 tima complex type 22 transport 13 treat complex type 22 transport 13 treat complex type 22 transport 13 treat complex type 23 titex tomplex type 23 titex tomplex type 24 tima complex type 24 tima complex type 24 tima complex type 24 tima complex type 25 tima complex type 26 message body 15 incoming multimedia message 15 incoming text me		
time complex type 21 thayout complex type 21 thest complex type 22 ther complex type 22 ther complex type 23 transport 13 transport 14 transport 13 transport 14 transport 13 transport 14		
Itayout complex type 21 IMMSSIIdes complex type 22 Itansport 13 IRealon complex type 23 IRealor complex type 23 ITo complex type 24 IXmsBat complex type 24 IXmsBat complex type 25 IXmsResponse complex type 26 Important type 27 IXmsData complex type 27 IXmsData complex type 28 IXmsResponse complex type 26 IXmsResponse complex type 27 IXmsResponse complex type 27 IXmsResponse complex type 28 IXmsResponse complex type 29 IXMsResponse complex type 29 IXMsResponse complex type 29 IXMsResponse complex type 20 IXMsResponse 20 IXMsResponse 20 IXMsResponse 20 IXMsResponse 20 IXMsResponse 30 IX		
IMMesta complex type 22 IPMar complex type 22 IPMar complex type 22 IPMar complex type 22 IPMar complex type 23 IRBodin complex type 23 ITD complex type 24 ILBer complex type 24 ILBer complex type 24 ILBer complex type 24 ILBer complex type 25 IXMsHeader complex type 26 IXMsHeader complex type 27 IXMsHeader complex type 28 IXMsHeader complex type 26 IXMsHeader complex type 27 IXMsHeader complex type 28 IXMsHeader complex type 26 IXMsHeader complex type 27 IXMsHeader complex type 27 IXMsHeader complex type 30 IXMsH		
IMMSSlides complex type 22 transport 13 Region complex type 22 transport 13 Region complex type 22 transport 13 Text complex type 23 tText complex type 23 tText complex type 24 tWinsbody complex type 24 tWinsbody complex type 25 tWinsteader complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming nuttimedia message 16 To 14 Subject 14 Sub		
Per complex type 22 transport 13 Realion complex type 23 Readino complex type 24 Root-layout complex type 24 Root layout complex type 25 Root layout lay	tMeta complex type 21	messages
Iransport 13 Region complex type 22 RequiredServiceTypeEnum simple type 26 Record complex type 23 Tio complex type 24 Tions type 24 Tions type 24 Tions type 24 Tions type 25 Tions type 24 Tions type 26 Tions type 27 Tions type 27 Tions type 28 Tions type 29 Tions type 20 Tions type 29 Tions 29 T	tMmsSlides complex type 22	DeliverXmsBatchSoapIn 41
Iransport 13 Region complex type 22 RequiredServiceTypeEnum simple type 26 Record complex type 23 Tio complex type 24 Tions type 24 Tions type 24 Tions type 24 Tions type 25 Tions type 24 Tions type 26 Tions type 27 Tions type 27 Tions type 28 Tions type 29 Tions type 20 Tions type 29 Tions 29 T		DeliverXmsBatchSoapOut 41
RequiredServiceTypeEnum simple type 26 Redict-layout complex type 23 Tfext complex type 24 Tfext complex type 24 Tfext complex type 24 Tfine complex type 25 Tfine complex type 26 Tfine complex type 27 Tfine complex type 27 Tfine complex type 28 Tfine complex type 28 Tfine complex type 28 Tfine complex type 29 Tfine complex type 29 Tfine complex type 29 Tfine complex type 29 Tfine complex type 20 Tfine complex type 21 Tfine complex type 21 Tfine complex type 24 Tfine complex type 26 Tfine complex type 27 Tfine complex type 28 Tfine complex type 27 Tfine complex type 28 Tfine complex type 28 Tfine complex type 29 Tfine complex type 29 Tfine complex type		
RequiredServiceTypeEnum simple type 26 RRoot-lavout complex type 23 Tiext complex type 23 Tiext complex type 24 TUSer complex type 24 TUSer complex type 24 TUSED complex type 24 TUSED complex type 25 TUSED complex type 25 TUSED complex type 25 TUSED complex type 25 TUSED complex type 26 Robile message packaged as MIME formatted e-mail message message body 15 Incoming ultimedia message 15 Incoming ultimedia message 15 Incoming text message 15 Incoming text message 15 TUSED content Class 14 To 15 TO 14 To 15 TO 1		
IRBot-layout complex type 23 Ticext complex type 24 TUSER complex type 25 TXMSBdat complex type 25 TXMSBdater complex type 25 TXMSBdater complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 Incoming multimedia message 15 Incoming text message 15 Incoming text message 15 Incoming multimedia message 15 Incoming text message 16 Incoming text message 15 Incoming text message 16 Incoming text message 10 Incoming text message 16 I		
treat complex type 24 ttloer complex type 24 ttlser complex type 24 ttmssDad complex type 24 txmsBody complex type 25 txmsReador complex type 25 txmsReador complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user-scenario 10 Obtaining information from the protocol server-scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsResponse 40 comps 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 DeliverXmsSoapOut 39 Simple types 40 DeliverXmsSoapOut 39 DeliverXmsSoapOut 30 Deli		
trio complex type 24 ttiser complex type 24 ttismsBody complex type 24 ttismsBody complex type 24 ttismsBody complex type 25 ttismsBader complex type 25 ttismsBeader complex type 25 ttismsBeader complex type 26 Mobile message packaged as MIME formatted e-mail message body 15 incoming multimedia message 15 incoming text message 15 incoming text message 15 incoming text message 16 incoming text message 17 Content-Class 14 tsmsBody complex type 26 Mobile message headers 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N		
tUser complex type 24 tXmsDaty complex type 25 tXmsHeader complex type 25 tXmsResponse complex type 25 tXmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N		
tXmsDada complex type 24 tXmsDada complex type 25 tXmsHeader complex type 25 tXmsHeader complex type 25 tXmsResponse complex type 25 tXmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message body 15 incoming multimedia message 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 Subject 14 Subject 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user – scenario 10 Oberations DeliverXms 38 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 dracoups 41 messages DeliverXmsSoapOut 39 simple types 40 dracoups 41 dracoups 41 messages DeliverXmsSoapOut 39 simple types 40 dracoups 41 dracoups 4		
tXmsData complex type 25 tXmsResponse complex type 25 tXmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 To 14 X-MS-Reply-To-Mobile 14 N N N N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server - scenario 10 Optaining information from the protocol server 9 Scenarios 10 AmsResponse 40 AmsRes	tUser complex type 24	tMMS SENDER 33
tXmsData complex type 25 tXmsResponse complex type 25 tXmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N	tXmsBody complex type 24	tServiceInfo 31
txmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming multimedia message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user—scenario 10 Obrations DeliverXms 38 attribute groups 41 attribute aroups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapOut 39 simple types 40 elements DeliverXmsSoapOut 39 Simple types 40 Obtaining information from an authenticated user of the protocol server of the pr		
tXmsResponse complex type 26 Mobile message packaged as MIME formatted e-mail message packaged as MIME formatted e-mail message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 from 14 X-MS-Reply-To-Mobile 14 N N N N N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user—scenario 10 Obtaining information from the protocol server—scenario 10 Operations DeliverXms 38 attribute aroups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 30 DeliverXms 3		
Mobile message packaged as MIME formatted e-mail message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N		
message message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N		
message body 15 incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N Namespaces 13 Normative references 8 OO Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server – scenario 10 Obtaining information from the protocol server operation DeliverXms 38 attributes 30 att		
incoming multimedia message 15 incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N	3	
incoming text message 15 message headers 14 Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N Namespaces 13 Normative references 8 O O Obtaining information from an authenticated user—scenario 10 Obtaining information from the protocol server—scenario 10 Obtaining information from the protocol server 9 DeliverXms 38 attributes 38 attribute groups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
message headers 14 Content-Class 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N		
Content-Class 14 From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N N N N N N N N N N N N N N N N N	incoming text message 15	messages
From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 Subject 14 X-MS-Reply-To-Mobile 14 SetUserInfo 34 Complex types tUserError 37 tXmsUser (section 3.1.4.2.3.1 37, section 3.1.4.2.3.2 37) elements O Subject 14 Complex types Subject 14 Complex types Subject 14 SetUserInfo 34 Complex types Subject 16 SetUserInfo 36 SetUserI	message headers 14	GetServiceInfoSoapIn 29
From 14 Subject 14 To 14 X-MS-Reply-To-Mobile 14 N N Namespaces 13 Normative references 8 O Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attribute aroups 41 attribute aroups 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXms 39 DeliverXms 39 DeliverXms 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 simple types 40 elements DeliverXmsSoapOut 39 simple types 40 obtaining information from an authenticated user 10 obtaining information from the protocol server	Content-Class 14	
Subject 14 To 14 X-MS-Reply-To-Mobile 14 Complex types tUserError 37 tXmsUser (section 3.1.4.2.3.1 37, section 3.1.4.2.3.2 37) elements Complex types tUserError 37 tXmsUser (section 3.1.4.2.3.1 37, section 3.1.4.2.3.2 37) elements GetUserInfo 36 GetUserInfo 36 GetUserInfo 36 GetUserInfo 36 MmsUser 36 Complex types 40 Elements DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXmsResponse 40 xmsResponse 40 xmsResponse 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 elements DeliverXmsSoapOut 39 simple types 40 elements DeliverXmsSoapOut 39 simple types 40 elements DeliverXmsSoapOut 39 simple types 40 data communication 10 obtaining information from the protocol server		
To 14 X-MS-Reply-To-Mobile 14 N N N Namespaces 13 Normative references 8 Normative references 8 O Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40		
X-MS-Reply-To-Mobile 14 N X-MS-Reply-To-Mobile 14 X-MS-Reply-To-Mobile 13 X-MS-User (Section 3.1.4.2.3.1 37, section 3.1.4.2.3.1 37, section 3.1.4.2.3.2 37) Reply-Self-Info 36 GetUserInfo 36 X-MS-User Info 36 GetUserInfo 36 GetUserInfo 36 X-MS-User Info 36 GetUserInfo 36 GetUserInfo 36 GetUserInfo 36 X-MS-User Info 36 GetUserInfo 36 GetUserInfo 36 X-MS-User Info 36 GetUserInfo 36 Messages GetUserInfo 36 X-MS-User Info 36 GetUserInfo 36 GetUserInfo 36 X-MS-User Info 36 GetUser Info 36 GetUser Info 36 X-MS-User Info 36 GetUser Info 36 X-MS-User Info 36 GetUser Info 36 Messages GetUser Info 36 Messages GetUser Info 36 X-MS-User Info 36 Messages GetUser Info 36 X-MS-User Info 36 Messages GetUser Info 4 Yes user Info 4		
tUserError 37 tXmsUser (section 3.1.4.2.3.1 37, section 3.1.4.2.3.1 37) elements O Obtaining information from an authenticated user - scenario 10 Obtaining information from the protocol server - scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXmsResponse 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapOut 39 DeliverXmsSoapOut 39 Simple types 40 obtaining information from the protocol server - scenario 10 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from an authenticated user		
tXmsUser (section 3.1.4.2.3.1 37, section 3.1.4.2.3.1 37, section 3.1.4.2.3.2 37) elements GetUserInfo 36 GetUserInfoResponse 36 userInfo 36 GetUserInfoResponse 36 userInfo 36 messages GetUserInfoSoapIn 35 GetUserInfoSoapIn 35 GetUserInfoSoapOut 35 operation DeliverXms 38 attribute groups 41 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 Simple types 40 elements DeliverXmsSoapOut 39 Simple types 40 data communication 10 obtaining information from an authenticated user	X-MS-Reply-10-Mobile 14	
Namespaces 13 Normative references 8 O O Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 qroups 41 messages DeliverXmsSoapOut 39 simple types 40 attribute aroups 38 attribute groups 38 attributes 38 qroups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from the protocol server		
Namespaces 13 Normative references 8 O O Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attribute groups 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 groups 41 messages DeliverXmsSoapOut 39 simple types 40 groups 41 messages DeliverXmsSoapOut 39 simple types 40 groups 41 messages DeliverXmsSoapOut 39 simple types 40 simple types 40 groups 41 messages DeliverXmsSoapOut 39 simple types 40	N	
Normative references 8 Obtaining information from an authenticated user—scenario 10 Obtaining information from the protocol server—scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms Response 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 protocol clients 10 protocol server 9 scenarios 10 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from the protocol server		
GetUserInfoResponse 36 userInfo 36 xmsUser 36 Obtaining information from an authenticated user - scenario 10 Obtaining information from the protocol server - scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 DeliverXmsSoapOut 39 simple types 40 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 Obtaining information from the protocol server 10 obtaining information from the protocol server	Namespaces 13	elements
GetUserInfoResponse 36 userInfo 36 xmsUser 36 Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapOut 39 simple types 40	Normative references 8	GetUserInfo 36
Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 DeliverXmsSoapOut 35 Operation attribute groups 38 attributes 38 qroups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		GetUserInfoResponse 36
xmsUser 36 Obtaining information from an authenticated user - scenario 10 Obtaining information from the protocol server - scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 obtaining information from an authenticated user GetUserInfoSoapIn 35 GetUserInfoSoapOut 35 operation attribute groups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server	^	
Obtaining information from an authenticated user – scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 arroups 41 messages DeliverXmsSoapOut 39 simple types 40 data communication 10 obtaining information from the protocol server setUserInfoSoapIn 35 GetUserInfoSoapOut 35 operation attribute groups 38 attributes 38 groups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 obtaining information from an authenticated user 10 obtaining information from the protocol server	U	
Scenario 10 Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 Attribute groups 41 Attributes 40 Complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 XmsData 40 XmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 35 Operation Attribute groups 38 Attributes 38 Attribute groups 48 Attributes 38 Attribu		
Obtaining information from the protocol server – scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapOut 39 peliverXmsSoapOut 39 simple types 40	·	
scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 Operation attribute groups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
scenario 10 Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 Operation attribute groups 38 attributes 38 groups 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server	Obtaining information from the protocol server –	
Operations DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 attributes 38 groups 38 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 pobtaining information from the protocol server		
DeliverXms 38 attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 attributes 38 groups 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		attribute groups 38
attribute groups 41 attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 attributes 40 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		attributes 38
attributes 40 complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 simple types 38 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
complex types 40 elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 qroups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 Simple types 40 Send reply message to client after collecting them from the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
elements DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 prom the recipient's phone 44 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user		
DeliverXms 39 DeliverXmsResponse 40 xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 Overview roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
DeliverXmsResponse 40 xmsData 40 xmsResponse 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 roles 9 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		
xmsData 40 xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 protocol clients 10 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user		
xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server	<u>DeliverXmsResponse</u> 40	
xmsResponse 40 groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40 protocol server 9 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server	xmsData 40	
groups 41 messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 Simple types 40 scenarios 10 data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server		protocol server 9
messages DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 Simple types 40 data communication 10 Obtaining information from an authenticated user 10 Obtaining information from the protocol server		
DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 Simple types 40 DeliverXmsSoapOut 39 DeliverXmsSoapOut 30 DeliverXmsSoapOut 30 DeliverXmsSoapOut 30 DeliverXmsSoapOut 30 DeliverXmsSoapOut 30 DeliverXmsSoapOut 30 DeliverXmsSoap	<u>9:04/3</u> ⊤±	Scenarios 10
DeliverXmsSoapOut 39 simple types 40 10 obtaining information from the protocol server	messages	
simple types 40 obtaining information from the protocol server		data communication 10
Simple Cybes 10	DeliverXmsSoapIn 39	data communication 10 obtaining information from an authenticated user
DeliverYmsRatch 41	<u>DeliverXmsSoapIn</u> 39 <u>DeliverXmsSoapOut</u> 39	data communication 10 obtaining information from an authenticated user 10
Deliver Attisbatch 41	DeliverXmsSoapIn 39 DeliverXmsSoapOut 39 simple types 40	data communication 10 obtaining information from an authenticated user 10 obtaining information from the protocol server

Overview (synopsis) 9	xmsResponses 43
_	groups 44
P	messages
B	<u>DeliverXmsBatchSoapIn</u> 41 DeliverXmsBatchSoapOut 41
Parameters - security index 51	simple types 44
Preconditions 11	GetServiceInfo operation 29
Prerequisites 11 Product behavior 59	attribute groups 34
Protocol clients – roles 10	attributes 34
Protocol Details	complex types
overview 28	tLONG SMS SENDER 33
Protocol server- roles 9	tMMS SENDER 33
Trotocor server roles	tServiceInfo 31
R	tSMS SENDER 32
K	tSupportedService 32
References 8	elements
informative 9	GetServiceInfo 30
normative 8	GetServiceInfoResponse 30
Relationship to other protocols 11	serviceInfo 30
Roles 9	groups 34
protocol clients 10	messages
protocol server 9	GetServiceInfoSoapIn 29
	GetServiceInfoSoapOut 30
S	simple types
	tAuthenticationTypeEnum 34
Scenarios 10	GetUserInfo operation 34
data communication 10	attribute groups 38
obtaining information from an authenticated user	attributes 38
10	complex types
obtaining information from the protocol server 10	typeller (section 3.1.4.3.3.1.37 section
Security	tXmsUser (<u>section 3.1.4.2.3.1</u> 37, <u>section</u> 3.1.4.2.3.2 37)
implementer considerations 51	elements
parameter index 51	GetUserInfo 36
Send reply message from protocol server to protocol	GetUserInfoResponse 36
client example 50	userInfo 36
Sequencing rules	xmsUser 36
client 45	groups 38
server 29 Server	messages
abstract data model 28	GetUserInfoSoapIn 35
DeliverXms operation 38	GetUserInfoSoapOut 35
attribute groups 41	simple types 38
attributes 40	<u>initialization</u> 28
complex types 40	<u>local events</u> 45
elements	message processing 29
DeliverXms 39	Send reply message to client after collecting them
DeliverXmsResponse 40	from the recipient's phone operation 44
xmsData 40	sequencing rules 29
xmsResponse 40	timer events 44
groups 41	timers 28
messages	Simple types 26
<u>DeliverXmsSoapIn</u> 39	tRequiredServiceTypeEnum 26 Standards assignments 12
DeliverXmsSoapOut 39	Syntax
simple types 40	messages - overview 13
DeliverXmsBatch operation 41	incoodges overview 15
attribute groups 44	Т
attributes 44	•
complex types	tAudio complex type 17
tXmsBatch 43	tBody complex type 17
tXmsDataInBatch 43	tContent complex type 17
tXmsResponseWithId 44	tDeliveryError complex type 18
elements DeliverXmsBatch 42	tHeader complex type 21
<u>DeliverXmsBatch</u> 42 <u>DeliverXmsBatchResponse</u> 42	Timer events
xmsBatch 42	client 45
ATTISDUCCII TZ	

```
server 44
Timers
 client 45
  server 28
tImg complex type 21
tLayout complex type 21
tMeta complex type 21
tMmsSlides complex type 22
tPar complex type 22
Tracking changes 60
Transport 13
tRegion complex type 22
tRequiredServiceTypeEnum simple type 26
tRoot-layout complex type 23
tText complex type 23
tTo complex type 24
tUser complex type 24
tXmsBody complex type 24
tXmsData complex type 25
tXmsHeader complex type 25
tXmsResponse complex type 26
Types
 complex 15
 simple 26
V
Vendor-extensible fields 12
Versioning 12
W
WSDL 52
```