

**Open Service Access (OSA);
Application Programming Interface (API);
Part 15: Multi-Media Messaging SCF
(Parlay 5)**



iTeh STANDARD PREVIEW
(standards.iteh.ai)
Full standard:
<https://standards.iteh.ai/catalog/standards/sist/4ed55144-a1e0-4848-9402-a5591d6783b3/etsi-es-203-915-15-v1.3.1-2008-04>



Reference

RES/TISPAN-01055-15-OSA

Keywords

API, IDL, OSA, UML

ETSI

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

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

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

Important notice

Individual copies of the present document can be downloaded from:

<http://www.etsi.org>

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at

<http://portal.etsi.org/tb/status/status.asp>

If you find errors in the present document, please send your comment to one of the following services:

http://portal.etsi.org/chaicor/ETSI_support.asp

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2008.

© The Parlay Group 2008.

All rights reserved.

DECT™, PLUGTESTS™, UMTS™, TIPHON™, the TIPHON logo and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members.

3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

Contents

Intellectual Property Rights	8
Foreword.....	8
1 Scope	9
2 References	9
3 Definitions and abbreviations.....	9
3.1 Definitions	9
3.2 Abbreviations	9
4 Multi Media Messaging SCF	10
5 Sequence Diagrams	10
5.1 Sending messages and receiving delivery notification	10
5.2 Sending, and receiving messages in same context	12
5.3 Setting notification of received messages	13
5.4 Using Mailbox functions	14
5.5 Using Mailbox to send and receive	16
5.6 Setting notification of received messages	18
6 Class Diagrams.....	19
7 The Service Interface Specifications	20
7.1 Interface Specification Format	20
7.1.1 Interface Class	20
7.1.2 Method descriptions.....	20
7.1.3 Parameter descriptions	20
7.1.4 State Model	20
7.2 Base Interface.....	20
7.2.1 Interface Class IpInterface	20
7.3 Service Interfaces	20
7.3.1 Overview	20
7.4 Generic Service Interface	21
7.4.1 Interface Class IpService	21
7.4.1.1 Method setCallback()	21
7.4.1.2 Method setCallbackWithSessionID().....	21
8 Multi Media Messaging Interface Classes	22
8.1 Interface Class IpMultiMediaMessagingManager	22
8.1.1 Method openMailbox()	23
8.1.2 Method openMultiMediaMessaging()	24
8.1.3 Method createNotification().....	25
8.1.4 Method destroyNotification()	25
8.1.5 Method changeNotification().....	25
8.1.6 Method getNextNotification()	26
8.1.7 Method enableNotifications()	27
8.1.8 Method disableNotifications()	27
8.2 Interface Class IpAppMultiMediaMessagingManager.....	27
8.2.1 Method mailboxTerminated()	28
8.2.2 Method reportNotification().....	28
8.2.3 Method notificationsInterrupted().....	29
8.2.4 Method notificationsResumed().....	29
8.2.5 Method multiMediaMessagingTerminated()	29
8.2.6 Method terminateMultipleMailboxes().....	29
8.2.7 Method terminateMultipleMultiMediaMessagingSessions().....	30
8.3 Interface Class IpMailbox	30
8.3.1 Method close()	31
8.3.2 Method createFolderReq()	32

8.3.3	Method getFoldersReq()	32
8.3.4	Method deleteFolderReq()	33
8.3.5	Method copyFolderReq()	33
8.3.6	Method moveFolderReq()	34
8.3.7	Method putMessageReq()	34
8.3.8	Method copyMessageReq()	35
8.3.9	Method moveMessageReq()	36
8.3.10	Method deleteMessageReq()	36
8.3.11	Method listMessagesReq()	37
8.3.12	Method listMessageBodyPartsReq()	37
8.3.13	Method getMessageBodyPartsReq()	38
8.3.14	Method getMessageHeadersReq()	39
8.3.15	Method getMessageContentReq()	39
8.3.16	Method getFullMessageReq()	40
8.3.17	Method getMailboxInfoPropertiesReq()	40
8.3.18	Method getFolderInfoPropertiesReq()	41
8.3.19	Method getMessageInfoPropertiesReq()	41
8.3.20	Method setMessageInfoPropertiesReq()	42
8.4	Interface Class IpAppMailbox	42
8.4.1	Method createFolderRes()	44
8.4.2	Method createFolderErr()	45
8.4.3	Method getFoldersRes()	45
8.4.4	Method getFoldersErr()	45
8.4.5	Method deleteFolderRes()	46
8.4.6	Method deleteFolderErr()	46
8.4.7	Method copyFolderRes()	46
8.4.8	Method copyFolderErr()	47
8.4.9	Method moveFolderRes()	47
8.4.10	Method moveFolderErr()	47
8.4.11	Method putMessageRes()	48
8.4.12	Method putMessageErr()	48
8.4.13	Method copyMessageRes()	48
8.4.14	Method copyMessageErr()	49
8.4.15	Method moveMessageRes()	49
8.4.16	Method moveMessageErr()	49
8.4.17	Method deleteMessageRes()	50
8.4.18	Method deleteMessageErr()	50
8.4.19	Method listMessagesRes()	50
8.4.20	Method listMessagesErr()	51
8.4.21	Method listMessageBodyPartsRes()	51
8.4.22	Method listMessageBodyPartsErr()	51
8.4.23	Method getMessageBodyPartsRes()	52
8.4.24	Method getMessageBodyPartsErr()	52
8.4.25	Method getMessageHeadersRes()	52
8.4.26	Method getMessageHeadersErr()	53
8.4.27	Method getMessageContentRes()	53
8.4.28	Method getMessageContentErr()	54
8.4.29	Method getFullMessageRes()	54
8.4.30	Method getFullMessageErr()	54
8.4.31	Method getMailboxInfoPropertiesRes()	55
8.4.32	Method getFolderInfoPropertiesRes()	55
8.4.33	Method getMessageInfoPropertiesRes()	55
8.4.34	Method setMessageInfoPropertiesRes()	56
8.4.35	Method setMessageInfoPropertiesErr()	56
8.4.36	Method getMailboxInfoPropertiesErr()	57
8.4.37	Method getFolderInfoPropertiesErr()	57
8.4.38	Method getMessageInfoPropertiesErr()	57
8.5	Interface Class IpMultiMediaMessaging	58
8.5.1	Method sendMessageReq()	58
8.5.2	Method cancelMessageReq()	59
8.5.3	Method queryStatusReq()	60
8.5.4	Method close()	60

8.5.5	Method <<new>> sendMessageWithNotifyReq()	61
8.6	Interface Class IpAppMultiMediaMessaging	62
8.6.1	Method sendMessageRes()	62
8.6.2	Method sendMessageErr()	63
8.6.3	Method cancelMessageRes()	63
8.6.4	Method cancelMessageErr()	63
8.6.5	Method queryStatusRes()	64
8.6.6	Method queryStatusErr()	64
8.6.7	Method messageStatusReport()	64
8.6.8	Method messageReceived()	65
8.6.9	Method <<new>> sendMessageWithNotifyRes()	65
8.6.10	Method <<new>> sendMessageWithNotifyErr()	66
9	State Transition Diagrams	66
10	Multi-Media Messaging Service Properties	67
11	Data Definitions	68
11.1	Multi-Media Messaging data definitions	68
11.1.1	IpMultiMediaMessagingManager	68
11.1.2	IpMultiMediaMessagingManagerRef	68
11.1.3	IpAppMultiMediaMessagingManager	68
11.1.4	IpAppMultiMediaMessagingManagerRef	68
11.1.5	IpMailbox	68
11.1.6	IpMailboxRef	68
11.1.7	IpAppMailbox	68
11.1.8	IpAppMailboxRef	68
11.1.9	IpMultiMediaMessaging	68
11.1.10	IpMultiMediaMessagingRef	68
11.1.11	IpAppMultiMediaMessaging	68
11.1.12	IpAppMultiMediaMessagingRef	69
11.1.13	TpBodyPartDescription	69
11.1.14	TpBodyPartDescriptionList	69
11.1.15	TpBodyPart	69
11.1.16	TpBodyPartList	69
11.1.17	TpDeliveryTime	69
11.1.18	TpDeliveryTimeType	70
11.1.19	TpFolderInfoProperty	70
11.1.20	TpFolderInfoPropertyName	70
11.1.21	TpFolderInfoPropertySet	70
11.1.22	TpGenericHeaderField	70
11.1.23	TpListMessagesCriteria	71
11.1.24	TpMailboxFolderStatusInformation	71
11.1.25	TpMailboxIdentifier	71
11.1.26	TpMailboxIdentifierSet	71
11.1.27	TpMailboxInfoProperty	71
11.1.28	TpMailboxInfoPropertyName	72
11.1.29	TpMailboxInfoPropertySet	72
11.1.30	TpMailboxMessageStatus	72
11.1.31	TpMessageDeliveryType	73
11.1.32	TpMessageInfoProperty	73
11.1.33	TpMessageInfoPropertyName	73
11.1.34	TpMessageInfoPropertySet	73
11.1.35	TpMessageHeaderFieldType	74
11.1.36	TpMessageHeaderField	75
11.1.37	TpMessageHeaderFieldSet	75
11.1.38	TpMessagePriority	75
11.1.39	TpMessageDeliveryReportType	76
11.1.40	TpMessageTreatment	76
11.1.41	TpMessageTreatmentType	76
11.1.42	TpMessageTreatmentSet	76
11.1.43	TpMultiMediaMessagingIdentifier	76
11.1.44	TpMultiMediaMessagingIdentifierSet	77

11.1.45	TpQueryStatusReport	77
11.1.46	TpQueryStatusReportSet	77
11.1.47	TpTerminatingAddressList	77
11.2	Event Notification data definitions	77
11.2.1	TpMessagingEventName	77
11.2.2	TpMessagingEventCriteria	77
11.2.3	TpMessagingEventCriteriaSet	78
11.2.4	TpNewMailboxMessageArrivedCriteria	78
11.2.5	TpNewMessageArrivedCriteria	78
11.2.6	TpMessagingEventInfo	78
11.2.7	TpMessagingEventInfoSet	79
11.2.8	TpNewMailboxMessageArrivedInfo	79
11.2.9	TpNewMessageArrivedInfo	79
11.2.10	TpMessageDescription	79
11.2.11	TpMessageDescriptionList	80
11.2.12	TpMessagingNotificationRequested	80
11.2.13	TpMessagingNotificationRequestedSet	80
11.2.14	TpMessagingNotificationRequestedSetEntry	80
11.2.15	TpNewMessageStatusReportArrivedInfo	80
11.3	Error type data definitions	81
11.3.1	TpMessageInfoPropertyError	81
11.3.2	TpMessagingError	81
11.3.3	TpMessageInfoPropertyErrorSet	82
11.3.4	TpSetPropertyError	82
12	Exception Classes	82
Annex A (normative):	OMG IDL Description of Multi-Media Messaging SCF	84
Annex B (informative):	W3C WSDL Description of Multi-Media Messaging SCF	85
Annex C (informative):	Java API Description of the Multi-Media Messaging SCF	86
Annex D (informative):	Contents of 3GPP OSA R6 Multi-Media Messaging	87
Annex E (informative):	Description of Multi Media Messaging for 3GPP2 cdma2000 networks	88
E.1	General Exceptions	88
E.2	Specific Exceptions	88
E.2.1	Clause 1: Scope	88
E.2.2	Clause 2: References	88
E.2.3	Clause 3: Definitions and abbreviations	88
E.2.4	Clause 4: Multi Media Messaging SCF	88
E.2.5	Clause 5: Sequence Diagrams	88
E.2.6	Clause 6: Class Diagrams	88
E.2.7	Clause 7: The Service Interface Specifications	88
E.2.8	Clause 8: Multi Media Messaging Interface Classes	89
E.2.9	Clause 9: State Transition Diagrams	89
E.2.10	Clause 10: Multi-Media Messaging Service Properties	89
E.2.11	Clause 11: Data Definitions	89
E.2.12	Clause 12: Exception Classes	89
E.2.13	Annex A (normative): OMG IDL Description of Multi-Media Messaging SCF	89
E.2.14	Annex B (informative): W3C WSDL Description of Multi-Media Messaging SCF	89
E.2.15	Annex C (informative): Java API Description of the Multi-Media Messaging SCF	89
Annex F (informative):	Record of changes	90
F.1	Interfaces	90
F.1.1	New	90
F.1.2	Deprecated	90
F.1.3	Removed	90

F.2	Methods.....	91
F.2.1	New.....	91
F.2.2	Deprecated.....	91
F.2.3	Modified.....	91
F.2.4	Removed.....	91
F.3	Data Definitions.....	92
F.3.1	New.....	92
F.3.2	Modified.....	92
F.3.3	Removed.....	92
F.4	Service Properties.....	92
F.4.1	New.....	92
F.4.2	Deprecated.....	93
F.4.3	Modified.....	93
F.4.4	Removed.....	93
F.5	Exceptions.....	93
F.5.1	New.....	93
F.5.2	Modified.....	94
F.5.3	Removed.....	94
F.6	Others.....	94
	History.....	95

iTeh STANDARD PREVIEW
 (standards.iteh.ai)

Full standard:
<https://standards.iteh.ai/catalog/standards/sist/4ed55144-a1eb-4848-9402-a5591d6783b3/etsi-es-203-915-15-v1.3.1-2008-04>

Intellectual Property Rights

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

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

Foreword

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

The present document is part 15 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 203 915) is structured in the following parts:

- Part 1: "Overview";
- Part 2: "Common Data Definitions";
- Part 3: "Framework";
- Part 4: "Call Control";
- Part 5: "User Interaction SCF";
- Part 6: "Mobility SCF";
- Part 7: "Terminal Capabilities SCF";
- Part 8: "Data Session Control SCF";
- Part 9: "Generic Messaging SCF";
- Part 10: "Connectivity Manager SCF";
- Part 11: "Account Management SCF";
- Part 12: "Charging SCF";
- Part 13: "Policy Management SCF";
- Part 14: "Presence and Availability Management SCF";
- Part 15: "Multi-Media Messaging SCF".**

The present document has been defined jointly between ETSI, The Parlay Group (<http://www.parlay.org>) and the 3GPP, in co-operation with a number of JAIN™ Community (<http://www.java.sun.com/products/jain>) member companies.

The present document forms part of the Parlay 5.2 set of specifications.

The present document is equivalent to 3GPP TS 29.198-15 V6.6.0 (Release 6).

1 Scope

The present document is part 15 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

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

The present document specifies the Multi Media Messaging Service Capability Feature (SCF) aspects of the interface. All aspects of the Multi Media Messaging SCF are defined here, these being:

- Sequence Diagrams.
- Class Diagrams.
- Interface specification plus detailed method descriptions.
- State Transition diagrams.
- Data Definitions.
- IDL Description of the interfaces.

The process by which this task is accomplished is through the use of object modelling techniques described by the Unified Modelling Language (UML).

2 References

The references listed in clause 2 of ES 203 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 203 915-1: "Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview (Parlay 5)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 203 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in ES 203 915-1 apply.

4 Multi Media Messaging SCF

The following clauses describe each aspect of the Multi Media Messaging Service Capability Feature (SCF).

The order is as follows:

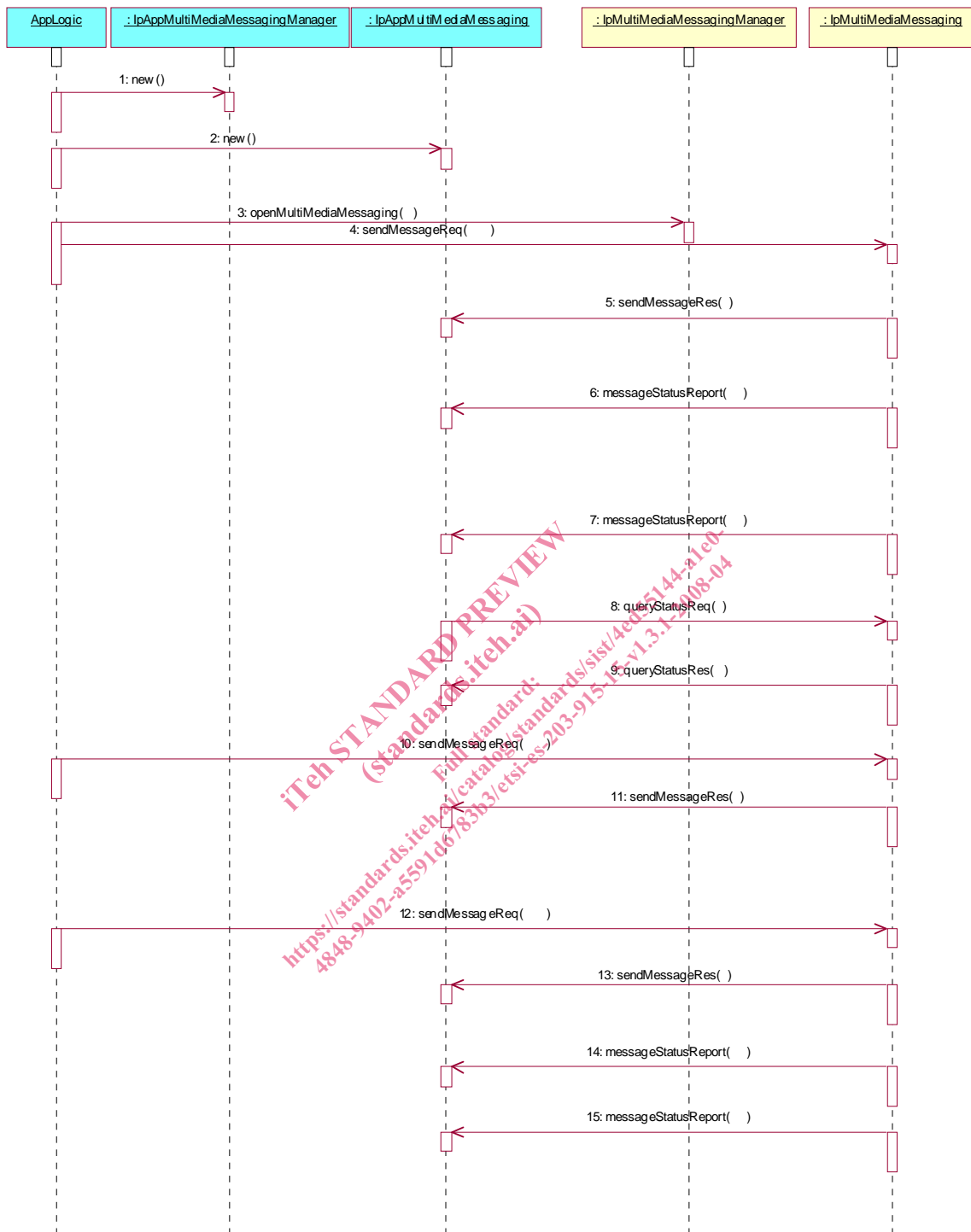
- The Sequence diagrams give the reader a practical idea of how each of the SCF is implemented.
- The Class relationships clause shows how each of the interfaces applicable to the SCF, relate to one another.
- The Interface specification clause describes in detail each of the interfaces shown within the Class diagram part.
- The State Transition Diagrams (STD) show the transition between states in the SCF. The states and transitions are well-defined; either methods specified in the Interface specification or events occurring in the underlying networks cause state transitions.
- The Data Definitions clause shows a detailed expansion of each of the data types associated with the methods within the classes. Note that some data types are used in other methods and classes and are therefore defined within the Common Data types part ES 203 915-2.

An implementation of this API which supports or implements a method described in the present document, shall support or implement the functionality described for that method, for at least one valid set of values for the parameters of that method. Where a method is not supported by an implementation of a Service interface, the exception P_METHOD_NOT_SUPPORTED shall be returned to any call of that method.

5 Sequence Diagrams

5.1 Sending messages and receiving delivery notification

This sequence diagram shows how the application can send messages on the IpMultiMediaMessaging interface with `sendMessageReq()`, and how the application can be informed about the delivery status of the message with `messageStatusReport()`. It also shows how the application can query the delivery status of a message, with `queryStatusReq()`.



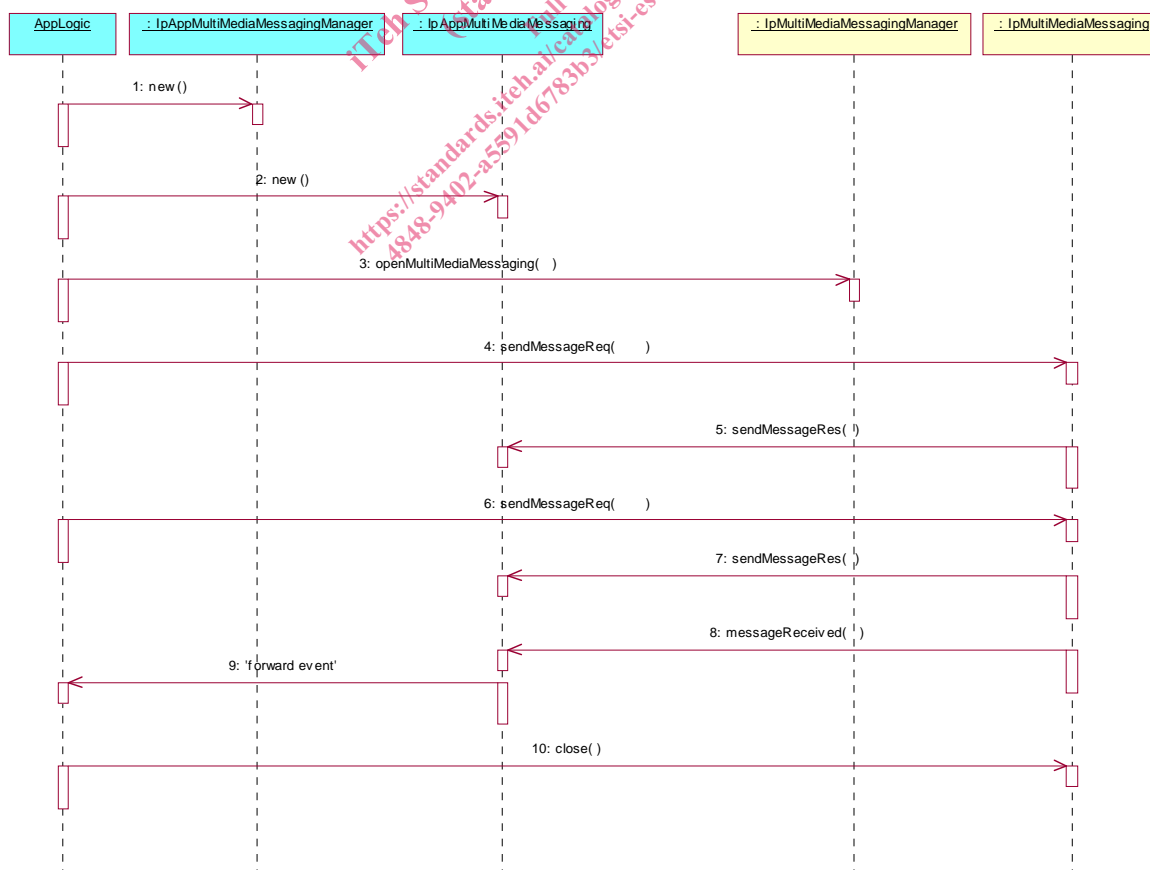
3: Request the opening of a MultiMedia Messaging object. The application intends to use this object to send messages to multiple destinations, so it has not specified any defaultDestinationAddressList.

4: The application sends a message. The destination address is included in the destinationAddressList parameter. If the source address was not provided when the IpMultiMediaMessaging object was created, it can be provided in the sourceAddress parameter. The application has requested delivery receipt and read receipt in the messageTreatment parameter. The assignmentID received as a return parameter enables the application to match any message status information with this message.

- 5: This method indicates successful processing of the sendMessageReq by the SCF, and that the message has been sent. It does not indicate a delivery status.
- 6: This method contains a delivery receipt for the message just sent.
- 7: This method contains a read receipt for the message just sent.
- 8: The application queries the status of the message it has sent (to verify the read receipt? or it has discarded the read receipt?).
- 9: The status of the message is returned.
- 10: The application sends another message, this time to a different destination. It has requested a read receipt to be returned.
- 11: This method indicates successful processing of the sendMessageReq by the SCF, and that the message has been sent. It does not indicate a delivery status.
- 12: The application sends another message, to a different destination. It has requested a read receipt to be returned.
- 14: This method contains an indication that the previous message has been read.
- 15: This method contains an indication that the second message has been read. The assignmentID is used to match this report to the corresponding sendMessageReq().

5.2 Sending, and receiving messages in same context

This sequence diagram shows how the application can send and receive messages within the same communication context using sendMessageReq() on the IpMultiMediaMessaging interface and messageReceived() on the IpAppMultiMediaMessaging interface.



3: Request the opening of a MultiMedia Messaging object. The application intends to use this object to send messages to the same destination, so it has specified the defaultdestinationAddressList. The defaultSourceAddress is also specified.

4: The application sends a message. The application has not included a destination address in the destinationAddressList parameter, as a default value has already been supplied in the openMultiMediaMessaging() method. Likewise the default source address was provided when the IpMultiMediaMessaging object was created, so there is no need to provide the sourceAddress parameter. The application has not requested delivery receipt or read receipt in the messageTreatment parameter.

5: This method indicates successful processing of the sendMessageReq by the SCF, and that the message has been sent. It does not indicate a delivery status.

6: The application sends another message to the same destination, again using default values for the destination and source addresses.

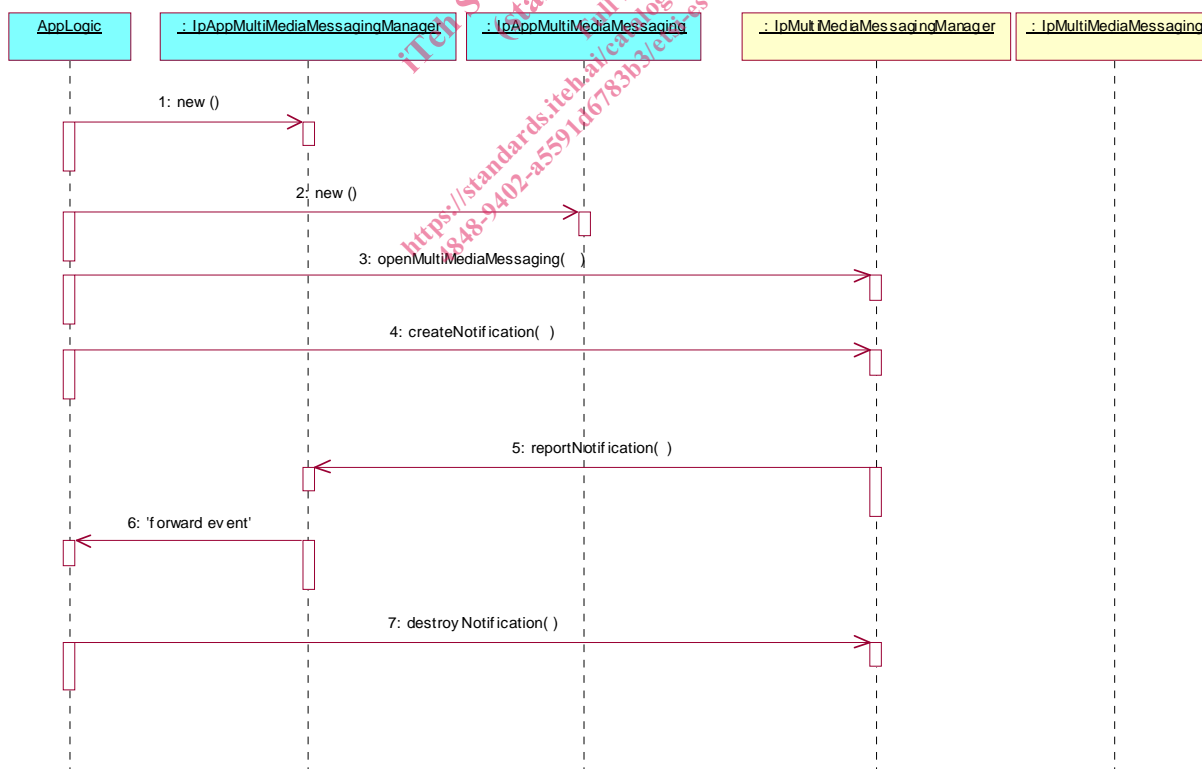
7: This method indicates successful processing of the sendMessageReq by the SCF, and that the message has been sent. It does not indicate a delivery status.

8: A new message is received in this communication context. The full message contents are carried in this method. It is not specified how the SCF identifies that this message is to be delivered in this communication context. The SCF could use source or destination addresses, content type, time or subject, among other parameters, to identify the context.

10: The application closes the session, i.e. closes the communication context.

5.3 Setting notification of received messages

This sequence diagram shows how the application can subscribe to notifications, and how it can receive messages using reportNotifications() method.



3: The application requests the opening of a MultiMedia Messaging object.