# [MS-OXOSMMS]: SMS and MMS Object Protocol Specification

## Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights**. This protocol 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 may make copies of it in order to develop implementations of the protocols, and may distribute portions of it in your implementations of the protocols or your documentation as necessary to properly document the implementation. This permission also applies to any documents that are referenced in the protocol documentation.

- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.

- **Patents**. Microsoft has patents that may cover your implementations of the protocols. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, the protocols may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp/default.mspx). If you would prefer a written license, or if the protocols are not covered by the OSP, patent licenses are available by contacting protocol@microsoft.com.

- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.

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

**Preliminary Documentation**. This documentation is preliminary documentation for these protocols. Since the documentation may change between this preliminary version and the final version, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

**Tools**. This protocol documentation is intended for use in conjunction with publicly available standard specifications and networking programming art, and assumes that the reader is either familiar with the aforementioned material or has immediate access to it. A protocol specification does not require the use of Microsoft programming tools or programming environments in order for a Licensee to develop an implementation. Licensees who have access to Microsoft programming tools and environments are free to take advantage of them.

| Revision Summary | | | |
|---|---|---|---|
| Author | Date | Version | Comments |
| Microsoft Corporation | April 4, 2008 | 0.1 | Initial Availability |

# Table of Contents

# 1 Introduction

This document specifies the SMS and MMS Object Protocol, which defines properties of objects that model Short Message Service (SMS) text messages and Multimedia Messaging Service (MMS) messages. SMS and MMS messages are delivered to other users via messaging protocols not specified in this document.

## 1.1 Glossary

The following terms are defined in [MS-OXGLOS]:

**folder object**
**GUID**
**handle**
**message object**
**named property**
**name ID or name identifier**
**property**
**property ID**
**special folder**
**Unicode**
**UTC**

The following terms are specific to this document:

**SMS**: Short Message Service, a communications protocol designed for text messages to be sent between mobile phones.

**SMS Object:** A **message object** that represents an **SMS** message in a messaging store and that adheres to the relevant property specifications in this document.

**MMS**: Multimedia Messaging Service, a communications protocol designed for messages containing text, images, and other multimedia content sent between mobile phones.

**MMS Object:** A **message object** that represents an **MMS** message in a messaging store and that adheres to the relevant property specifications in this document.

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

## 1.2 References

### 1.2.1 Normative References

[MS-OXCFOLD] Microsoft Corporation, "Folder Object Protocol Specification", April 2008.

[MS-OXCMAIL] Microsoft Corporation, "RFC2822 and MIME to E-mail Object Conversion Protocol Specification", April 2008.

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol Specification", April 2008.

[MS-OXCPRPT] Microsoft Corporation, "Property and Stream Object Protocol Specification", April 2008.

[MS-OXGLOS] Microsoft Corporation, "Office Exchange Protocols Master Glossary", April 2008.

[MS-OXOMSG] Microsoft Corporation, "E-mail Object Protocol Specification", April 2008.

[MS-OXOSFLD] Microsoft Corporation, "Special Folders Protocol Specification", April 2008.

[MS-OXPROPS] Microsoft Corporation, "Office Exchange Protocols Master Property List Specification", April 2008.

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

### 1.2.2 Informative References

[SMIL] W3C, Michel, T., "Synchronized Multimedia", March 2008, http://www.w3.org/AudioVideo/.

## 1.3 Protocol Overview (Synopsis)

The SMS and MMS Object Protocol specifies the representation of SMS text messages and MMS multimedia messages in a messaging store. This protocol extends the Message and Attachment Object Protocol in that it defines new properties and adds restrictions to the properties that are defined in [MS-OXCMSG].

This document specifies the properties that are unique to SMS objects and MMS objects. An SMS object is characterized by a short unformatted text body. An MMS object is characterized by text and multimedia components. SMS and MMS objects are stored in **folder objects**. The SMS and MMS Object Protocol also specifies how an SMS or MMS object is created and manipulated.

## 1.4 Relationship to Other Protocols

The SMS and MMS Object Protocol has the same dependencies as the Message and Attachment Object Protocol, which it extends. For details about the Message and Attachment Object Protocol, see [MS-OXCMSG].

The SMS and MMS Object Protocol is a peer of the E-mail Object Protocol, and uses a subset of the properties specified in [MS-OXOMSG].

## 1.5 Prerequisites/Preconditions

The SMS and MMS Object Protocol has the same prerequisites and preconditions as the Message and Attachment Object Protocol.

## 1.6 Applicability Statement

None.

## 1.7 Versioning and Capability Negotiation

None.

## 1.8 Vendor-Extensible Fields

This protocol provides no vendor-extensibility beyond what is already specified in [MS-OXCMSG].

## 1.9 Standards Assignments

None.

# 2 Messages

## 2.1 Transport

The SMS and MMS Object Protocol uses the protocols defined in [MS-OXCPRPT] and [MS-OXCMSG] as its primary transport mechanism.

## 2.2 Message Syntax

SMS and MMS objects can be created and modified by clients and servers. Except where noted below, this section defines constraints under which both clients and servers operate.

Clients operate on SMS and MMS objects using the Message and Attachment Object Protocol, as specified in [MS-OXCMSG]. How a server operates on SMS and MMS objects is implementation-dependent. The results of any such operations are exposed to clients in a manner that is consistent with the SMS and MMS Object Protocol.

Unless otherwise specified below, SMS and MMS objects adhere to all property constraints specified in [MS-OXPROPS] and all property constraints specified in [MS-OXCMSG]. SMS and MMS objects MAY also contain other properties, which are defined in [MS-OXPROPS], but these properties have no impact on the SMS and MMS Object Protocol.

### 2.2.1 Common SMS and MMS object properties

#### 2.2.1.1 PidLidOMSAccountGuid

Type: PtypString.

Encodes the GUID of the OMS account used to deliver the message in the following format (including the braces): {DWORD-WORD-WORD-WORD-WORD.DWORD}, for example, “{c200e360-38c5-11ce-ae62-08002b2b79ef}”.

#### 2.2.1.2 PidLidOMSScheduleTime

Type: PtypTime, in UTC.

The time that the client requested the service provider send the SMS or MMS message.

#### 2.2.1.3 PidLidOMSServiceType

Type: PtypInteger32.

Indicates the type of service used to send the SMS or MMS message; MUST be one of the following:

| Value | Meaning |
|---|---|
| 0x00000001 | SMS |
| 0x00000004 | MMS |

#### 2.2.1.4 PidLidOMSSourceType

Type: PtypInteger32.

Indicates the source of the SMS or MMS message; MUST be one of the following:

| Value | Source Type |
|---|---|
| 0x00000000 | XMS Inspector |
| 0x00000001 | Reminder |
| 0x00000002 | Calendar Summary |
| 0x00000003 | Rule |
| 0x00000004 | Unknown |

#### 2.2.1.5 PidNameContentClass

Type: PtypString.

Set on an SMS or MMS object according to [MS-OXCMAIL].

| Value | Meaning |
|---|---|
| “MS-OMS-SMS” | SMS |
| “MS-OMS-MMS” | MMS |

### 2.2.2 Additional Property Constraints

This protocol specifies additional constraints on the following properties beyond what is specified in [MS-OXCMSG] and [MS-OXOMSG].

#### 2.2.2.1 PidTagIconIndex

Type:  PtypInteger32.

Specifies which icon is to be used by a user interface when displaying a group of SMS and/or MMS objects; SHOULD be set <1>; if set, MUST be 0xFFFFFFFF.

#### 2.2.2.2 PidTagMessageClass

Type: PtypString8, case-insensitive.

Specifies the type of the message object. In addition to meeting the criteria specified in [MS-OXCMSG], MUST be “`IPM.Note.Mobile.SMS`” or begin with “`IPM.Note.Mobile.SMS.`” for SMS objects; MUST be “`IPM.Note.Mobile.MMS`” or begin with “`IPM.Note.Mobile.MMS.`” for MMS objects.

#### 2.2.2.3 Body Properties

The contents of SMS message objects are stored and retrieved following the plain text body specification in [MS-OXCMSG] <2>.

The contents of MMS message objects are stored and retrieved following the HTML body specification in [MS-OXCMSG] <3>.

#### 2.2.2.4 PidTagNormalizedSubject

Type: PtypString.

Contains an abbreviated version of the contents of the message suitable for displaying groups of SMS objects to a user. For MMS objects, only the constraints in [MS-OXCMSG] apply.

# 3 Protocol Details

General protocol details, as specified in [MS-OXPROPS] and [MS-OXCMSG], apply.

## 3.1 Common Details

The client and server roles are to create and operate on SMS and MMS objects, and otherwise operate in their roles as specified in [MS-OXCMSG].

### 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.

#### 3.1.1.1 Folders

An SMS or MMS object is created in the Drafts, Inbox or Sent Items **special folder** as defined in [MS-OXOSFLD] unless the end-user or user agent explicitly specifies another **folder object**.

### 3.1.2 Timers

None.

### 3.1.3 Initialization

None.

### 3.1.4 Higher-Layer Triggered Events

#### 3.1.4.1 Creation of an SMS or MMS Object

To create an SMS or MMS object, the server or client sets properties in accordance with the requirements in section 2 and [MS-OXCPRPT], and saves the resulting message object as specified in [MS-OXCMSG].

#### 3.1.4.2 Modification of an SMS or MMS Object

When modifying an SMS or MMS object, the client or server modifies any of the properties in accordance with the requirements in section 2 and [MS-OXCPRPT], and saves the message object as specified in [MS-OXCMSG].

#### 3.1.4.3 Deletion of an SMS or MMS Object

An SMS or MMS object has no special deletion semantics beyond what is defined in [MS-OXCFOLD].

### 3.1.5 Message Processing Events and Sequencing Rules

None.

### 3.1.6 Timer Events

None.

### 3.1.7 Other Local Events

None.

# 4 Protocol Examples

## 4.1 Sample SMS Object

Joe creates an SMS object, types in some text, and sends it. The following is a description of what a client might do to accomplish Joe's intentions and the responses a server might return. See [MS-OXCPRPT] and [MS-OXCMSG] for details on ROPs.

Before manipulating SMS objects, the client needs to ask the server to perform a mapping from **named properties** to **property IDs**, using RopGetPropertyIdsFromNames:

| Property | Property Set GUID | NameID |
|---|---|---|
| pidlidOmsMobileModel | {00020329-0000-0000-C000000000046} | "OMSMobileModel" |
| pidlidOmsAccountGuid | {00020329-0000-0000-C000000000046} | "OMSAccountGuid" |
| pidlidOmsServiceType | {00020329-0000-0000-C000000000046} | "OMSServiceType" |
| pidlidOmsSourceType | {00020329-0000-0000-C000000000046} | "OMSSourceType" |

The server might respond with the following identifiers, which will be used in the example that follows. (The actual identifiers are at the discretion of the server.)

| Property | Property ID |
|---|---|
| pidlidOmsMobileModel | 0x84c3 |
| pidlidOmsAccountGuid | 0x84c4 |
| pidlidOmsServiceType | 0x84c5 |
| pidlidOmsSourceType | 0x84c6 |

To create an SMS object, the client uses **RopCreateMessage**. The server returns a success code and a **handle** to a message object.

After Joe has input his content for the SMS object, the client uses **RopSetProperties** to transmit his data to the server.

| **Property** | **Property ID** | **Data Type** | **Value** |
|---|---|---|---|
| PidLidOmsAccountGuid | 0x84c4 | 0x001f (PtypString) | "{01234567-0123-0123-0123-0123456789ab}" |
| PidLidOmsMobileModel | 0x84c3 | 0x001f (PtypString) | (null) |
| PidLidOmsServiceType | 0x84c5 | 0x0003 (PtypInteger32) | 0x00000001 |
| PidLidOmsSourceType | 0x84c6 | 0x0003 (PtypInteger32) | 0x00000000 |
| PidTagBody | 0x1000 | 0x001f (PtypString) | "What time is the meeting?" |
| PidTagInternetCpid | 0x3fde | 0x0003 (PtypInteger32) | 0x0000FDE9 |

| | | | |
|---|---|---|---|
| PidTagMessageClass | 0x001a | 0x001e (PtypString8) | "IPM.Note.Mobile.SMS" |
| PidTagNormalizedSubject | 0x0e1d | 0x001f (PtypString) | "What time is the meeting?" |
| PidTagSubjectPrefix | 0x003d | 0x001f (PtypString) | (null) |

When Joe is ready to send his message, the client uses RopSaveChangesMessage to commit the properties on the server, and then RopRelease to release the SMS object. The client then submits the message to an SMS provider using an appropriate messaging protocol.

The values of some properties will change during the execution of RopSaveChangesMessage, but the properties specified in [MS-OXOSMMS] will not change.

## 4.2 Sample MMS Object

Joe creates an MMS object, gives it a subject, types in some text, attaches a picture, and sends it. The following is a description of what a client might do to accomplish Joe’s intentions and the responses a server might return. See [MS-OXCPRPT] and [MS-OXCMSG] for details on ROPs.

Before manipulating an MMS object, the client needs to ask the server to perform a mapping from **named properties** to **property IDs**, using RopGetPropertyIdsFromNames:

| Property | Property Set GUID | NameID |
|---|---|---|
| pidlidOmsMobileModel | {00020329-0000-0000-C000000000046} | "OMSMobileModel" |
| pidlidOmsAccountGuid | {00020329-0000-0000-C000000000046} | "OMSAccountGuid" |
| pidlidOmsServiceType | {00020329-0000-0000-C000000000046} | "OMSServiceType" |
| pidlidOmsSourceType | {00020329-0000-0000-C000000000046} | "OMSSourceType" |

The server might respond with the following identifiers, which will be used in the example that follows. (The actual identifiers are at the discretion of the server.)

| Property | Property ID |
|---|---|
| pidlidOmsMobileModel | 0x84ce |
| pidlidOmsAccountGuid | 0x84cf |
| pidlidOmsServiceType | 0x84d0 |
| pidlidOmsSourceType | 0x84d1 |

To create an MMS object, the client uses RopCreateMessage. The server returns a success code and a handle to an object.

After Joe has input his content for the MMS object, the client uses RopSetProperties to transmit his data to the server.

| Property | Property ID | Data Type | Value |
|---|---|---|---|
| PidLidOmsAccountGuid | 0x84cf | 0x001f (PtypString) | “{01234567-0123-0123-0123456789abc}” |
| PidLidOmsMobileModel | 0x84ce | 0x001f (PtypString) | (empty) |
| PidLidOmsServiceType | 0x84d0 | 0x0003 (PtypInteger32) | 0x00000004 |
| PidLidOmsSourceType | 0x84d1 | 0x0003 (PtypInteger32) | 0x00000000 |

| | | | |
|---|---|---|---|
| PidTagInternetCpid | 0x3fde | 0x0003 (PtypInteger32) | 0x0000FDE9 |
| PidTagHtml | 0x1013 | 0x0102 (PtypBinary) | See below |
| PidTagIconIndex | 0x1080 | 0x0003 (PtypInteger32) | 0xFFFFFFFFFF |
| PidTagMessageClass | 0x001a | 0x001e (PtypString8) | "IPM.Note.Mobile.MMS" |
| PidTagMessageFlags | 0x0e07 | 0x0003 (PtypInteger32) | Flags: 0x00000018<br>MSGFLAG_UNSENT<br>MSGFLAG_HASATTACH |
| PidTagNormalizedSubject | 0x0e1d | 0x001f (PtypString) | "Here's the photo." |
| PidTagSubjectPrefix | 0x003d | 0x001f (PtypString) | (empty) |

PidTagHtml is a binary property containing the following text:

```
<HTML>
<BODY>
<IMG SRC="cid:Att1.jpg@AB1B43B2B0594564.B94EF7ABB12B49BA"
border="0">
<BR>
This is the photo you asked for.
<BR>
<A HREF="cid:Att0.txt@AB1B43B2B0594564.B94EF7ABB12B49BA"></A>
</BODY>
</HTML>
```

The client uses RopCreateAttachment to allocate space for a data file in the message. The server returns a success code and a handle to an attachment object. The client then uses this handle with RopSetProperties to transmit data about the attachment to the server.

| Property | Property ID | Data Type | Value |
|---|---|---|---|
| PidTagAttachmentHidden | 0x7ffe | 0x000b (PtypBoolean) | 0x01 |
| PidTagAttachMethod | 0x3705 | 0x0003 (PtypInteger32) | 0x00000001 (ATTACH_BY_VALUE) |
| PidTagAttachContentId | 0x3712 | 0x001f (PtypString) | "mms.smil@AB1B43B2B0594564.B94EF7ABB12B49BA" |
| PidTagAttachMimeTag | 0x370e | 0x001f (PtypString) | "application/smil" |
| PidTagAttachLongFilename | 0x3707 | 0x001f (PtypString) | "mms.smil" |

The client sets the contents of the attachment by using the attachment handle with RopOpenStream, passing in PidTagAttachDataBin as the property to open. With the handle returned from RopOpenStream, the client calls RopWriteStream, writing out the contents of the Synchronized Multimedia Integration Language (SMIL) file, the format of which is detailed in [SMIL], describing the layout of the MMS message. The client follows this with

RopRelease on the stream handle, then RopSaveChangesAttachment to commit the changes, and RopRelease to release the handle to the attachment.

The client repeats the process from RopCreateAttachment to RopRelease with the attachment handle twice more, once for a plain-text version of the body, and once for the image. The attachment containing the body uses the following properties and values with RopSetProperties:

| Property | Property ID | Data Type | Value |
|---|---|---|---|
| PidTagAttachmentHidden | 0x7ffe | 0x000b<br>(PtypBoolean) | 0x01 |
| PidTagAttachMethod | 0x3705 | 0x0003<br>(PtypInteger32) | 0x00000001<br>(ATTACH_BY_VALUE) |
| PidTagAttachContentId | 0x3712 | 0x001f<br>(PtypString) | "Att0.txt@AB1B43B2B0594564.<br>B94EF7ABB12B49BA" |
| PidTagAttachMimeTag | 0x370e | 0x001f<br>(PtypString) | "text/plain" |
| PidTagAttachLongFilename | 0x3707 | 0x001f<br>(PtypString) | "1.txt" |

The RopOpenStream for the plain-text body is also on PidTagAttachDataBin, but the contents written are Unicode text. The last attachment the client creates contains the image, and the RopSetProperties sends the following data:

| Property | Property ID | Data Type | Value |
|---|---|---|---|
| PidTagAttachmentHidden | 0x7ffe | 0x000b<br>(PtypBoolean) | 0x01 |
| PidTagAttachMethod | 0x3705 | 0x0003<br>(PtypInteger32) | 0x00000001<br>(ATTACH_BY_VALUE) |
| PidTagAttachContentId | 0x3712 | 0x001f<br>(PtypString) | "Att1.jpg@AB1B43B2B0594564.<br>B94EF7ABB12B49BA" |
| PidTagAttachMimeTag | 0x370e | 0x001f<br>(PtypString) | "image/jpeg" |
| PidTagAttachLongFilename | 0x3707 | 0x001f<br>(PtypString) | "photo.jpg" |

The contents of PidTagAttachDataBin on the image attachment are the binary contents of the image file.

When Joe is ready to send his message, the client uses RopSaveChangesMessage to commit the properties on the server, and then RopRelease to release the MMS object. The client then submits the message to an MMS provider using an appropriate messaging protocol.

The values of some properties will change during the execution of RopSaveChangesMessage, but the properties specified in in this protocol will not change.

# 5 Security

## 5.1 Security Considerations for Implementers

There are no special security considerations specific to the SMS and MMS Object Protocol. General security considerations pertaining to the underlying transport apply, as specified in [MS-OXCMSG] and [MS-OXCPRPT].

## 5.2 Index of Security Parameters

None.

# 6 Appendix A: Office / Exchange Behavior

The information in this specification is applicable to the following versions of Office/Exchange:

- Microsoft Exchange 2003 with Service Pack 2 applied
- Microsoft Office 2007 with Service Pack 1 applied
- Microsoft Exchange 2007 with Service Pack 1 applied

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Office/Exchange behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies Office/Exchange does not follow the prescription.

<1> Section 2.2.2.1: "Microsoft Office Outlook 2007" does not always set the PidTagIconIndex property on SMS or MMS objects.

<2> Section 2.2.2.3: "Microsoft Office Outlook 2007" sets both PidTagBody and PidTagHtml on SMS objects.

<3> Section 2.2.2.3: "Microsoft Office Outlook 2007" sets both PidTagBody and PidTagHtml on MMS objects.

# 7 Index