# [MS-OXCMAIL]:
# RFC2822 and MIME to E-Mail Object Conversion Protocol Specification

---

**Intellectual Property Rights Notice for Open Specifications Documentation**

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **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 technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@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.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

---

## Revision Summary

| Date | Revision History | Revision Class | Comments |
|---|---|---|---|
| 04/04/2008 | 1.0.0 | Major | Initial Availability. |
| 04/25/2008 | 0.2 | Editorial | Revised and updated property names and other technical content. |
| 06/27/2008 | 1.0 | Major | Initial Release. |
| 08/06/2008 | 1.01 | Editorial | Revised and edited technical content. |
| 09/03/2008 | 1.02 | Editorial | Revised and edited technical content. |
| 12/03/2008 | 1.03 | Editorial | Revised and edited technical content. |
| 02/04/2009 | 1.04 | Editorial | Revised and edited technical content. |
| 03/04/2009 | 1.05 | Editorial | Revised and edited technical content. |
| 04/10/2009 | 2.0 | Major | Updated technical content and applicable product releases. |
| 07/15/2009 | 3.0 | Editorial | Revised and edited for technical content. |
| 11/04/2009 | 4.0.0 | Major | Updated and revised the technical content. |
| 02/10/2010 | 5.0.0 | Major | Updated and revised the technical content. |
| 05/05/2010 | 6.0.0 | Major | Updated and revised the technical content. |
| 08/04/2010 | 7.0 | Major | Significantly changed the technical content. |
| 11/03/2010 | 8.0 | Major | Significantly changed the technical content. |

# Contents


**1 Introduction ..... 7**
1.1 Glossary ..... 7
1.2 References ..... 9
1.2.1 Normative References ..... 9
1.2.2 Informative References ..... 12
1.3 Overview ..... 12
1.3.1 Data Models ..... 12
1.4 Relationship to Protocols and Other Structures ..... 13
1.5 Applicability Statement ..... 13
1.6 Versioning and Localization ..... 14
1.7 Vendor-Extensible Fields ..... 14

**2 Structures ..... 15**
2.1 MIME Generation ..... 15
2.1.1 Address Elements ..... 16
2.1.1.1 Recipients ..... 16
2.1.1.1.1 To and Cc Recipients ..... 18
2.1.1.1.2 Bcc recipients ..... 18
2.1.1.2 Reply-to ..... 18
2.1.1.3 From ..... 19
2.1.1.4 Sender ..... 19
2.1.1.5 Return Receipt ..... 19
2.1.1.6 Read Receipt ..... 20
2.1.1.7 Directory Lookups ..... 20
2.1.1.8 IMCEA Encapsulation ..... 20
2.1.1.9 PidTagAddressType ..... 21
2.1.2 Envelope Elements ..... 21
2.1.2.1 Message Class ..... 22
2.1.2.2 Content Class ..... 23
2.1.2.3 Unified Messaging Properties ..... 23
2.1.2.4 Arbitrary MIME Headers ..... 24
2.1.2.5 Importance ..... 24
2.1.2.6 Sensitivity ..... 25
2.1.2.7 Sent Time ..... 25
2.1.2.8 Subject ..... 25
2.1.2.9 Conversation Topic ..... 26
2.1.2.10 Conversation Index ..... 26
2.1.2.11 Message ID ..... 26
2.1.2.12 References ..... 26
2.1.2.13 Categories ..... 26
2.1.2.14 In-Reply-To Message ID ..... 26
2.1.2.15 List Server Properties ..... 27
2.1.2.16 Language Properties ..... 27
2.1.2.17 Classification Properties ..... 27
2.1.2.18 Payload Properties ..... 28
2.1.2.19 Has Attach ..... 28
2.1.2.20 Auto Response Suppress ..... 28
2.1.2.21 Is Auto Forwarded ..... 29
2.1.2.22 Sender Id Status ..... 29
2.1.2.23 Purported Sender Domain ..... 29

2.1.2.24 Spam Confidence Level ..... 30
2.1.2.25 Flag Request ..... 30
2.1.2.26 TNEF Correlation Key ..... 30
2.1.2.27 Received Headers ..... 30
2.1.2.28 ReplyBy Time ..... 30
2.1.2.29 Content-ID ..... 30
2.1.2.30 XRef ..... 31
2.1.3 Body Text ..... 31
2.1.3.1 Client Actions ..... 31
2.1.3.2 Message Body in TNEF ..... 32
2.1.3.3 Simple Plain Text Message Body ..... 33
2.1.3.4 HTML Text Message Body Without Inline Attachments ..... 33
2.1.3.5 HTML Text Message Body from RTF Without Inline Attachments ..... 33
2.1.3.6 HTML Text Message Body with Inline Attachments ..... 34
2.1.3.7 HTML Text Message Body from RTF with Inline (OLE) Attachments ..... 34
2.1.3.8 Calendar Items and Meeting Messages ..... 34
2.1.3.8.1 Plain Text Calendar Message ..... 34
2.1.3.8.2 Calendar Message Without Inline Attachments ..... 35
2.1.3.8.3 Calendar Message with Inline Attachments ..... 35
2.1.4 Attachments ..... 35
2.1.4.1 Inline Attachments ..... 37
2.1.4.1.1 Inline Attachments in RTF Messages ..... 37
2.1.4.1.2 Inline Attachments in HTML Messages ..... 37
2.1.4.2 Attached Files ..... 37
2.1.4.2.1 File Name ..... 38
2.1.4.2.2 Content-Type, Content-Description, Content-Disposition ..... 38
2.1.4.2.3 Content-ID, Content-Location, Content-Base ..... 39
2.1.4.2.4 Content-Transfer-Encoding, MIME Part Body ..... 39
2.1.4.3 MacBinary Attached Files ..... 39
2.1.4.4 OLE Attachments ..... 41
2.1.4.5 Embedded Message Attachments ..... 42
2.1.4.6 vCard Generation ..... 42
2.1.5 Generating Pure MIME Messages ..... 42
2.1.5.1 Generation Process ..... 43
2.2 MIME Analysis ..... 43
2.2.1 Address Elements ..... 44
2.2.1.1 Mapping Internet E-Mail Address Elements to a Property Group ..... 44
2.2.1.2 Recognizing and De-Encapsulating IMCEA-Encapsulated Addresses ..... 45
2.2.1.3 From ..... 45
2.2.1.4 Sender ..... 45
2.2.1.5 To, Cc, Bcc ..... 46
2.2.1.6 Reply Recipients ..... 46
2.2.1.7 Disposition Notification Recipients ..... 46
2.2.1.8 Return-Receipt-To ..... 47
2.2.2 Envelope Elements ..... 47
2.2.2.1 MessageID ..... 47
2.2.2.2 Sent time ..... 47
2.2.2.3 References ..... 48
2.2.2.4 Sensitivity ..... 48
2.2.2.5 Importance ..... 48
2.2.2.6 Subject ..... 49
2.2.2.6.1 Normalizing the Subject ..... 49
2.2.2.7 Conversation Topic ..... 49

2.2.2.8 Conversation Index ..... 50
2.2.2.9 In-Reply-To Message ID ..... 50
2.2.2.10 ReplyBy Time ..... 50
2.2.2.11 Language Properties ..... 50
2.2.2.12 Categories ..... 50
2.2.2.13 Message Expiry Time ..... 51
2.2.2.14 Suppression of Automatic Replies ..... 51
2.2.2.15 Content Class ..... 51
2.2.2.16 Message Flagging ..... 52
2.2.2.17 List Server Properties ..... 53
2.2.2.18 Payload Properties ..... 53
2.2.2.19 Purported Sender Domain ..... 53
2.2.2.20 Sender Id Status ..... 53
2.2.2.21 Spam Confidence Level ..... 54
2.2.2.22 Classification Properties ..... 54
2.2.2.23 Unified Messaging Properties ..... 54
2.2.2.24 Content-ID ..... 55
2.2.2.25 Content-Base ..... 55
2.2.2.26 Content-Location ..... 55
2.2.2.27 XRef ..... 55
2.2.2.28 PidTagTransportMessageHeaders ..... 55
2.2.2.29 Generic Headers in PS_INTERNET_HEADERS ..... 56
2.2.3 Body Text ..... 57
2.2.3.1 Client Actions ..... 57
2.2.3.2 Determining Which MIME Element Is the Message Body ..... 58
2.2.3.2.1 Selecting the Primary Message Text MIME Element ..... 59
2.2.4 Attachments ..... 59
2.2.4.1 Regular File Attachment MIME Part Analysis ..... 60
2.2.4.1.1 File name ..... 60
2.2.4.1.2 Content Type ..... 62
2.2.4.1.3 Attachment Creation and Modification Date ..... 62
2.2.4.1.4 Attachment Content-Id, Content-Base, and Content-Location ..... 62
2.2.4.1.5 Attachment Content-Transfer-Encoding and MIME Part Body ..... 63
2.2.4.2 Apple File Formats ..... 63
2.2.4.2.1 Multipart/Appledouble ..... 64
2.2.4.2.2 Application/Applefile ..... 65
2.2.4.2.3 Application/Mac-binhex40 ..... 68
2.2.4.3 Attached Messages ..... 69
2.2.4.4 Inbound vCard Conversion ..... 70
2.2.4.4.1 Content-Type ..... 71
2.2.4.4.2 General Parsing Guidelines ..... 71
2.2.5 External Body Attachments ..... 71
2.2.6 Reading Pure MIME Messages ..... 72
2.3 Additional Content Types ..... 72
2.3.1 Analysis of Non-MIME Content ..... 72
2.3.2 Message/Partial ..... 73
2.3.3 Multipart/Digest ..... 73
2.4 Preserving Unconverted MIME Parts on MIME Messages ..... 73
2.4.1 PidTagMimeSkeleton ..... 74
2.4.2 Impact of Message Changes on the MIME Skeleton ..... 76
2.4.3 MIME Conversion ..... 78

**3 Structure Examples ..... 80**

3.1 MIME Examples ..... 81
3.1.1 MIME Message Containing Inline and Non-Inline Attachments ..... 81
3.1.2 MIME Message Containing Only Inline Attachments ..... 82
3.1.3 MIME Message Containing Only Non-Inline Attachments ..... 83

**4 Security Considerations ..... 84**
4.1 Unsolicited Commercial E-mail (Spam) ..... 84
4.2 Information Disclosure ..... 84
4.3 Content-Type Versus File Extension Mismatch ..... 84
4.4 Do Not Support Message/Partial ..... 85
4.5 Considerations for Message/External-Body ..... 85
4.6 Preventing Denial of Service Attacks ..... 85
4.6.1 Submission Limits ..... 85
4.6.2 Complexity of Nested Entities ..... 86
4.6.3 Number of Embedded Messages ..... 86
4.6.4 Compressed Attachments ..... 86

**5 Appendix A: Product Behavior ..... 87**

**6 Change Tracking ..... 99**

**7 Index ..... 108**

# 1 Introduction

The [RFC2822] and **MIME** to **E-mail object** Conversion protocol specifies what clients and servers do when they have data in one of these formats, but need it in the other. The process of converting **Message object** data to MIME format is referred to as **MIME generation**, while the reverse process is referred to as **MIME analysis**.

## 1.1 Glossary

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

**.jpg**
**8.3 name**
**address book**
**address list**
**ASCII**
**attachment**
**Attachment object**
**attachments table**
**base64 encoding**
**best body**
**big-endian**
**binary large object (BLOB)**
**Bcc recipient**
**body part**
**calendar**
**Calendar object**
**Cc recipient**
**character set**
**charset**
**code page**
**contact**
**Contact object**
**Coordinated Universal Time (UTC)**
**distinguished name (DN)**
**domain**
**E-mail object**
**EntryID**
**flags**
**GUID**
**header**
**Hypertext Markup Language (HTML)**
**Internet Message Access Protocol – Version 4 (IMAP4)**
**locale**
**Mail User Agent (MUA)**
**mailbox**
**meeting**
**message**
**message body**
**message class**
**Message object**
**Messaging Application Programming Interface (MAPI)**
**metafile**
**MIME**

**MIME content-type**
**MIME entity**
**MIME message**
**named property**
**non-delivery report (NDR)**
**one-off address**
**one-off EntryID**
**Out of Office (OOF)**
**Personal Information Manager (PIM)**
**plain text**
**plain text message body**
**Post Office Protocol – Version 3 (POP3)**
**property (1)**
**property name**
**property type**
**recipient (2)**
**recipient table**
**reminder**
**restriction**
**Rich Text Format (RTF)**
**remote procedure call (RPC)**
**Simple Mail Transfer Protocol (SMTP)**
**S/MIME**
**spam**
**stream (1)**
**To recipient**
**top-level message**
**Transport Neutral Encapsulation Format (TNEF)**
**Unicode**
**universal unique identifier (UUID)**
**Unified messaging**
**Uniform Resource Identifier (URI)**
**UTF-16LE (Unicode Transformation Format, 16-bits, Little-Endian)**
**vCard**

The following terms are specific to this document:

**addressee property group:** A group of four related **properties** – display name, **EntryID**, e-mail **address type**, and e-mail address – that together specify one addressee on a **Message object**.

**contact attachment:** An attached **message** item with a **message** type of "IPM.Contact" that adheres to the definition of a **Contact object** described in [MS-OXOCNTC].

**delivery status notification (DSN):** A type of e-mail message, described in [RFC3464], that reports the result of an attempt to deliver a message to one or more recipients.

**Internet Mail Connector Encapsulated Address (IMCEA):** A means of encapsulating an e-mail address that is not compliant with [RFC2821] within an e-mail address that is compliant with [RFC2821].

**MIME analysis:** The process of converting data from an Internet wire protocol to a format suitable for **storage** by a server or a client.

**MIME body:** The content of a **MIME** entity, which follows the **header** of the **MIME** entity to which they both belong.

**MIME generation:** The process of converting data held by a server or a client to a format that is suitable for Internet-standard wire protocols.

**MIME reader:** An agent that performs **MIME** analysis; it might be either a client or server.

**MIME writer:** An agent that performs **MIME** generation; it might be either client or server.

**MUIDEMSAB:** A 128-bit quantity that prefixes an **EntryID** and identifies it as an Exchange **EntryID**.

**MUIDOOP:** A 128-bit quantity that prefixes an **EntryID** and identifies it as a one-off **entry ID**.

**primary SMTP proxy address:** The **SMTP** email address to be used to designate a **message** server user in all **SMTP** traffic. Proxy addresses are stored in the user's **address book** entry, in the multi-valued string **property** PidTagAddressBookProxyAddresses. The primary **SMTP** proxy address can be identified by its **address type** field, which is set to "**SMTP**" (all upper case). Non-primary **SMTP** proxy addresses have the address type field set to "smtp" (all lower case).

**PS_INTERNET_HEADERS:** An extensible namespace for custom **property** headers.

**pure MIME message:** A **MIME** representation of an e-mail **message** with no **Transport Neutral Encapsulation Format (TNEF) body part**.

**TNEF message:** A MIME representation of an e-mail **message** in which **attachments** and some **message properties** are carried in a **Transport Neutral Encapsulation Format (TNEF) body part**.

**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

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[IEEE1003.1] The Open Group, "IEEE Std 1003.1, 2004 Edition", 2004, http://www.unix.org/version3/ieee_std.html

[MS-DTYP] Microsoft Corporation, "Windows Data Types", March 2007, http://msdn.microsoft.com/en-us/library/cc230273.aspx

[MS-LCID] Microsoft Corporation, "Windows Language Code Identifier (LCID) Reference", March 2007, http://msdn.microsoft.com/en-us/library/cc233965.aspx

[MS-OXBBODY] Microsoft Corporation, "Best Body Retrieval Protocol Specification", April 2008.

[MS-OXCDATA] Microsoft Corporation, "Data Structures", April 2008.

[MS-OXCICAL] Microsoft Corporation, "iCalendar to Appointment Object Conversion Protocol Specification", April 2008.

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

[MS-OXCROPS] Microsoft Corporation, "Remote Operations (ROP) List and Encoding Protocol Specification", April 2008.

[MS-OXCSPAM] Microsoft Corporation, "Spam Confidence Level Protocol Specification", April 2008.

[MS-OXOABK] Microsoft Corporation, "Address Book Object Protocol Specification", April 2008.

[MS-OXOCAL] Microsoft Corporation, "Appointment and Meeting Object Protocol Specification", April 2008.

[MS-OXOCNTC] Microsoft Corporation, "Contact Object Protocol Specification", April 2008.

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

[MS-OXOPOST] Microsoft Corporation, "Post Object Protocol Specification", April 2008.

[MS-OXOSMIME] Microsoft Corporation, "S/MIME E-Mail Object Protocol Specification", April 2008.

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

[MS-OXPROTO] Microsoft Corporation, "Exchange Server Protocols System Overview", April 2008.

[MS-OXRTFEX] Microsoft Corporation, "Rich Text Format (RTF) Extensions Specification", April 2008.

[MS-OXTNEF] Microsoft Corporation, "Transport Neutral Encapsulation Format (TNEF) Data Structure", April 2008.

[MS-OXVCARD] Microsoft Corporation, "vCard to Contact Object Conversion Protocol Specification", April 2008.

[MS-WMF] Microsoft Corporation, "Windows Metafile Format Specification", June 2007, http://msdn.microsoft.com/en-us/library/cc215212.aspx

[MSFT-RTF] Microsoft Corporation, "Word 2007: Rich Text Format (RTF) Specification, version 1.9.1", March 2008, http://www.microsoft.com/downloads/details.aspx?FamilyID=DD422B8D-FF06-4207-B476-6B5396A18A2B&displaylang=en

[RFC1738] Berners-Lee, T., Masinter, L., and McCahill, M., "Uniform Resource Locators (URL)", RFC 1738, December 1994, http://www.ietf.org/rfc/rfc1738.txt

[RFC1740] Faltstrom, P., Crocker, D., and Fair, E., "MIME Encapsulation of Macintosh files – MacMIME", RFC 1740, December 1994, ftp://ftp.rfc-editor.org/in-notes/rfc1740.txt

[RFC1741] Faltstrom, P., Crocker, D., and Fair, E., "MIME Content Type for BinHex Encoded Files", RFC 1741, December 1994, ftp://ftp.rfc-editor.org/in-notes/rfc1741.txt

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, http://www.ietf.org/rfc/rfc2045.txt

[RFC2046] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996, http://www.ietf.org/rfc/rfc2046.txt

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996, http://www.ietf.org/rfc/rfc2047.txt

[RFC2049] Freed, N., and Borenstein N., "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996, ftp://ftp.rfc-editor.org/in-notes/rfc2049.txt

[RFC2076] Palme, J., "Common Internet Message Headers", RFC 2076, February 1997, ftp://ftp.rfc-editor.org/in-notes/rfc2076.txt

[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

[RFC2156] Kille, S., "MIXER (Mime Internet X.400 Enhanced Relay): Mapping between X.400 and RFC 822/MIME", RFC 2156, January 1998, http://www.ietf.org/rfc/rfc2156.txt

[RFC2369] Neufeld, G., and Baer, J., "The Use of URLs as Meta-Syntax for Core Mail List Commands and their Transport through Message Header Fields", RFC 2369, July 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2369.txt

[RFC2426] Dawson, F., and Howes, T., "vCard MIME Directory Profile", RFC 2426, September 1998, http://www.ietf.org/rfc/rfc2426.txt

[RFC2445] Dawson, F., and Stenerson, D., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", RFC 2445, November 1998, http://www.ietf.org/rfc/rfc2445.txt

[RFC2557] Palme, J., Hopmann, A., and Shelness, N., "MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2557, March 1999, http://www.ietf.org/rfc/rfc2557.txt

[RFC2821] Klensin, J., Ed., "Simple Mail Transfer Protocol", RFC 2821, April 2001, http://www.ietf.org/rfc/rfc2821.txt

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

[RFC2387] Levinson, E., "The MIME Multipart/Related Content-type", RFC 2387, August 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2387.txt

[RFC3030] Vaudreuil, G., "SMTP Service Extensions for Transmission of Large and Binary MIME Messages", RFC 3030, December 2000, http://www.rfc-editor.org/rfc/rfc3030.txt

[RFC3282] Alvestrand, H., "Content Language Headers", RFC 3282, May 2002, ftp://ftp.rfc-editor.org/in-notes/rfc3282.txt

[RFC3464] Moore, K., and Vaudreuil, G., "An Extensible Message Format for Delivery Status Notifications", RFC 3464, January 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3464.txt

[RFC3516] Nerenberg, L., "IMAP4 Binary Content Extension", RFC 3516, April 2003, http://www.rfc-editor.org/rfc/rfc3516.txt

[RFC3798] Hansen, T., and Vaudreuil, G., Eds., "Message Disposition Notification", RFC 3798, May 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3798.txt

[RFC4646] Phillips, A., and Davis, M., Eds., "Tags for Identifying Languages", RFC 4646, September 2006, http://www.ietf.org/rfc/rfc4646.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.ietf.org/rfc/rfc5234.txt

[RFC5751] Ramsdell, B., and Turner, S., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010, http://www.rfc-editor.org/rfc/rfc5751.txt

[UNICODE] The Unicode Consortium, "Unicode Home Page", http://www.unicode.org/

### 1.2.2 Informative References

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

[RFC1521] Borenstein, N., and Freed, N., "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, September 1993, ftp://ftp.rfc-editor.org/in-notes/rfc1521.txt

[X25] International Telecommunication Union, "Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit", ITU-T Recommendation X.25, October 1996, http://www.itu.int/rec/T-REC-X.25-199610-I/en

## 1.3 Overview

The representation of electronic mail, calendar items, and other **Personal Information Manager (PIM)** objects by message objects and their **properties** is described in [MS-OXPROTO] and detailed in [MS-OXOMSG], [MS-OXOCAL], and related specifications.

In contrast, electronic mail, calendar items, and other PIM objects are represented as textual **streams** when sent over Internet protocols. The textual representation of these streams is commonly referred to as RFC2822 and/or MIME format, as specified by [RFC2822], and [RFC2045] through [RFC2049].

The RFC2822 and MIME to E-mail Object Conversion protocol specifies how to convert between message objects and MIME-formatted textual streams. The process of converting message object data to MIME-formatted textual streams is referred to as MIME generation, while the reverse process is referred to as MIME analysis. Similarly, the agent that performs MIME generation (which might be either a client or server) is referred to as a **MIME writer**, and the agent that performs MIME analysis is referred to as a **MIME reader**.

### 1.3.1 Data Models

Message objects model e-mail **messages** and other PIM objects after a business memo: there is a single **message body**, with zero or more **attachments** and zero or more **recipients**. Each Message object has a **message class** property that indicates its type, and an arbitrary collection of properties. Attached messages allow for the nesting of content.

MIME, in contrast, models e-mail messages as a nested set of **MIME entities**, each of which has **headers** and a (possibly empty) body. No entity is distinguished as the message body. The Content-Type header indicates the type of each **body part**; other headers indicate whether a body part is intended as a message body or an attachment. Recipients are modeled by e-mail addresses in certain headers on the top-level body part. Multipart body parts allow for grouping and nesting of content, including attached messages.

The following table shows, at a high level, how the parts of each data model correspond.

| MIME | Message object |
|---|---|
| E-mail address | Recipient |
| Header | Property |
| Body part | Message body |
| Body part | Attachment |

At the next level of detail, some problems become apparent. Because the data models do not match exactly, it becomes more difficult to convert lower-level items between the Message object and MIME formats.

One of the challenges in mapping the Message object content to MIME comes from the need to generate human-readable text. A Message object property can be of a data type, such as a **binary large object (BLOB)**, that does not lend itself to representation as text. Two solutions are available for these problems:

1. Generate a pure MIME message, in which data that does not lend itself to representation in MIME is simply omitted from the MIME representation.
2. Generate a **Transport Neutral Encapsulation Format (TNEF)** message, in which data that does not lend itself to representation in MIME is placed in a TNEF body part with a Content-Type of application/ms-TNEF.

Challenges in mapping MIME content to Message objects include distinguishing message body from attachments; analyzing multi-part structures that do not fit the Message object data model; and mapping headers or header parameters that do not have any corresponding property.

Each Message object has a single **charset** (although nested Message objects can have different charsets). MIME, on the other hand, permits the charset of each header and message body to be specified separately.

## 1.4 Relationship to Protocols and Other Structures

Data on the MIME side of the conversion is specified by [RFC2822], [RFC2045] through [RFC2049], and related specifications as listed in section 1.2.1 or as referenced from the specifications themselves. Data on the message object side of the conversion is specified by [MS-OXCMSG], [MS-OXOMSG], [MS-OXOCAL], and related specifications as listed in [MS-OXPROTO].

## 1.5 Applicability Statement

Conversion between MIME and message object format is performed in the context of several different protocols. For example:

- Clients and servers perform MIME generation for mail outbound to **Simple Mail Transfer Protocol (SMTP)**.
- Servers perform MIME analysis for mail inbound from SMTP.
- Servers perform MIME generation for message objects that are downloaded via **Post Office Protocol – Version 3 (POP3)** or **Internet Message Access Protocol – Version 4 (IMAP4)**. Clients perform MIME generation for messages that are uploaded via IMAP4.

- Servers perform MIME analysis for message objects that are uploaded via IMAP4. Clients perform MIME analysis for Message objects that are downloaded via POP3 or IMAP4.

## 1.6 Versioning and Localization

This document covers localization issues in the following area:

- Localization: Localization-dependent content is specified in section 2.1.3 and section 2.2.3.

Localization is supported by marking messages with **locale** and **character set** information.

## 1.7 Vendor-Extensible Fields

[RFC2045] and related RFCs define extensibility mechanisms for MIME headers and content types.

[MS-OXPROPS] defines extensibility mechanisms for message object properties and message classes.

# 2 Structures

The bulk of this specification is divided into two parts. Section 2.1 specifies how MIME writers set message object properties to produce MIME data.<1> Section 2.2 specifies how MIME readers create MIME to produce a message object property or structure.

A wide variety of possible structures exist for MIME messages. One particular structure carries a Transport Neutral Encapsulation Format (TNEF) MIME element, which provides a high level of fidelity to original message object content. All **TNEF messages** have the same structure, as follows:

- At the top level, a MIME entity with a Content-Type of "multipart/mixed" that specifies all address elements, as well as the following two child entities:
  - A MIME entity with a Content-Type of "text/plain", that contains a **plain text** rendering of the message body.
  - A MIME entity with a Content-Type of "application/ms-tnef" and that contains all attachment content, the **Hypertext Markup Language (HTML)** or **Rich Text Format (RTF)** message body, and any message object properties for which no mapping to MIME headers is defined, encoded as specified in [MS-OXTNEF].

Because a TNEF message is a MIME structure, **MIME messages** without a TNEF element are sometimes referred to as "pure MIME" to distinguish them from TNEF messages.

## 2.1 MIME Generation

This section specifies both conversion to pure MIME and conversion to TNEF from message objects.

When generating a MIME rendering of a message object, whether pure MIME or TNEF, MIME writers retrieve all properties of the message object by issuing one of the following **ROP** sequences (see [MS-OXCROPS]):

- RopGetPropertiesList followed by RopGetPropertiesSpecific
- RopGetPropertiesAll

Clients can explicitly request conversion to pure MIME or TNEF by doing one of the following; a MIME writer SHOULD honor such a client request for message format:

- A client can request conversion to pure MIME for all recipients by setting the value of the PidTagSendRichInfo property to FALSE on the message object, and request conversion to TNEF for all recipients by setting the same property value to TRUE.
- A client can request conversion to pure MIME for an individual recipient by setting the value of the PidTagSendRichInfo property to FALSE on that recipient, and request conversion to TNEF for an individual recipient by setting the same property value to TRUE.
- A client can request conversion to pure MIME for a message sent using a **one-off address** by setting the **M** bit in the **one-off EntryID**, as specified in [MS-OXCDATA] section 2.2.5.1, and request conversion to TNEF by resetting the same bit.

Similarly, when conversion to pure MIME is requested, clients can explicitly request plain text or HTML message body generation by one of the following means; again, a MIME writer SHOULD honor such a client request for message format:

- A client can request a specific **MIME body** format for all recipients by setting the value of the PidTagSendInternetEncoding property on the message object.

- A client can request a specific MIME body format for an individual recipient by setting the value of the PidTagSendInternetEncoding property on the recipient.

The value of the PidTagSendInternetEncoding property is specified in the following table. The first value is the hexadecimal representation of the property value; the second is its representation in Advanced Backus-Naur Form (ABNF), as specified by [RFC5234].

| PidTagSendInternetEncoding property value (hex, ABNF) | Requested format |
|---|---|
| 0x00060000<br>%x00.00.06.00 | Plain text only |
| 0x000E0000,<br>%x00.00.0E.00 | HTML only <2> |
| 0x00160000,<br>%x00.00.16.00 | Both plain text and HTML |

### 2.1.1 Address Elements

In general, address elements are generated in pure MIME only and not in TNEF. However, when a TNEF message is generated, all address elements of messages that are included as attachments of the **top-level message** are generated in TNEF only, as specified in [MS-OXTNEF]. This is because there is no MIME entity that corresponds to the attached messages; the attached messages are wholly contained in the TNEF.

MIME writers MUST generate e-mail addresses for MIME recipients in compliance with the address requirements specified in [RFC2822]. For example, in cases where a display name is generated in a MIME address header, servers use the encoding specified by [RFC2047] to encode any display name value that has characters that are not allowed in a MIME header per [RFC2822].<3> These addresses are always SMTP addresses. When a client supports other types of e-mail addresses through the PidTagAddressType property, servers SHOULD use **Internet Mail Connector Encapsulated Address (IMCEA)** encapsulation of the e-mail address to form an SMTP address, as specified in section 2.1.1.8.<4>

Address elements other than recipients, such as From and Sender, are represented in a message object by an **addressee property group** of four properties: display name, address type, e-mail address, and **EntryID**. In subsequent sections, such properties might be referred to as a group. For example, "The PidTagSentRepresenting property group" includes the following properties: PidTagSentRepresentingName, PidTagSentRepresentingAddressType, PidTagSentRepresentingEmailAddress, and PidTagSentRepresentingEntryId.

#### 2.1.1.1 Recipients

To create a recipient in a MIME recipient header, clients create a message object recipient with either a PidTagEntryId property or both the PidTagAddressType and PidTagEmailAddress properties, which suffice to fully represent the recipient's e-mail address type and e-mail address. Clients SHOULD, in addition, set PidTagSmtpAddress, particularly to save the SMTP address when the value of PidTagAddressType is not SMTP.

Clients MUST set the PidTagRecipientType property value for each recipient as specified by the following table to indicate whether a recipient is a **To recipient**, a **Cc recipient**, or a **Bcc recipient**.

| PidTagRecipientType value | Recipient header |
|---|---|
| 0x00000001 | To |
| 0x00000002 | Cc |
| 0x00000003 | Bcc |

When generating MIME or TNEF, MIME writers SHOULD ignore recipient types other than To, Cc, and Bcc. MIME writers generate one MIME recipient for a Message object recipient that has a value of To, Cc, or Bcc.<5>Each MIME recipient MUST be generated in the header that corresponds to the PidTagRecipientType property value, as specified by the PidTagRecipientType value table.

Clients SHOULD set the PidTagDisplayName property for recipients, where that information is available. MIME writers SHOULD copy the PidTagDisplayName property value, when it exists, when generating the display name of an [RFC2822] address specification. If the PidTagDisplayName property is not assigned, then MIME writers SHOULD NOT generate a value for the display name. MIME writers can encode the display name, as specified in [RFC2047], in order to preserve non-ASCII characters.

MIME writers SHOULD generate the angle-address portion (angle-addr) of an ([RFC2822] section 3.4) address specification from addressee properties, used in the following order of preference: PidTagEntryId, PidTagAddressType / PidTagEmailAddress, PidTagSmtpAddress.<6>More specifically, MIME writers SHOULD do the following:

1. If PidTagEntryId is present and bytes 4-19 are equal to the **MUIDEMSAB universal unique identifier (UUID)** value "{%xdc.a7.40.c8.c0.42.10.1a.b4.b9.08.00.2b.2f.e1.82}", it is an **address book** EntryID. In this case, MIME writers SHOULD look up the address book entry that corresponds to the **distinguished name (DN)** that is contained in the EntryID, and use the **primary SMTP proxy address** that is found on the address book entry. (EntryID format is specified in [MS-OXCDATA] section 2.2, and the procedure for looking up address book entries is specified in [MS-OXOABK] section 2.1.)

2. Otherwise, if PidTagEntryId is present and bytes 4-19 are equal to the **MUIDOOP** UUID value "{%x81.2b.1f.a4.be.a3.10.19.9d.6e.00.dd.01.0f.54.02}", it is a one-off EntryID. The e-mail type and address are encoded in the EntryID, as specified in [MS-OXCDATA]. If the e-mail type is SMTP, use this e-mail address; otherwise, continue to step 6.

3. If PidTagEntryId is present and bytes 4-19 are some value other than the values that are shown in steps 1 and 2, the MIME writer rejects the recipient. If MIME generation is being done for SMTP, a failure Delivery Status Notification (DSN) is generated for that recipient. The format of a failure DSN is specified in [RFC3464]. The corresponding Message object structure is referred to as a **non-delivery report** and is specified in [MS-OXOMSG] section 2.2.2.

4. Otherwise, if both PidTagAddressType and PidTagEmailAddress are present and PidTagAddressType matches SMTP, use the value of PidTagEmailAddress.

5. Otherwise, if the PidTagSmtpAddress property is present, use its value.

6. Otherwise, if an e-mail address and **address type** are present, whether obtained from PidTagAddressType and PidTagEmailAddress or from an EntryID, but the address type does not match SMTP, the MIME writer SHOULD attempt IMCEA encapsulation of the e-mail address, as specified in section 2.1.1.8.

7. Finally, if all of the previous conditions fail, the MIME writer MUST reject the recipient. If MIME generation is being done for outbound SMTP, a failure DSN is generated for that recipient. The

format of a failure DSN is specified in [RFC3464]. The corresponding message object structure is referred to as a non-delivery receipt; its format is specified in [MS-OXOMSG].

#### 2.1.1.1.1 To and Cc Recipients

To generate a To or Cc MIME header, clients add a recipient to the message object and set the PidTagRecipientType property to the value that corresponds to the individual recipient type, as specified by the PidTagRecipientType value table in section 2.1.1.1.

MIME writers map recipients to the To or Cc MIME headers as requested by clients. An exceptional situation occurs when generating MIME for an attached **delivery status notification (DSN)** message. A DSN message is one that has the following PidTagMessageClass value:

```
; The most common values are "REPORT.IPM.Note.NDR" and "REPORT.IPM.Note.DR"
ReportMsgClass = "REPORT" 1*("." MsgClassToken) (".NDR" / ".DR")
MsgClassToken = ALPHA *(ALPHA / DIGIT)
```

In that case, MIME writers ignore the recipients of the attached message and instead populate the To header of the attached message by using the PidTagReceivedRepresenting property group of the attached message. If the PidTagReceivedRepresenting property group is not defined, MIME writers use the PidTagReceivedBy property group of the attached message.

When generating TNEF, MIME writers MUST NOT generate attRecipTable for the **top-level message**.<7> For attached messages, MIME writers MUST copy all recipients, along with all their properties, into the attRecipTable TNEF attribute in the TNEF body part, as specified in [MS-OXTNEF]. This applies to attached DSN messages as well.

#### 2.1.1.1.2 Bcc recipients

To generate a Bcc MIME header, clients add a recipient to the message object and set the PidTagRecipientType property value for that recipient to "0x00000003".

When generating a message for outbound submission to SMTP, MIME writers MUST NOT copy Bcc recipients to the MIME Bcc header except for **meeting** and task messages. This also applies to the MIME Bcc header of attached messages. MIME writers MUST NOT copy Bcc recipients to the TNEF attRecipTable for attached messages.

When generating a message for protocols such as POP3 and IMAP4, MIME writers SHOULD copy Bcc recipients to the MIME Bcc header. This also applies to the MIME Bcc header of the attached messages. MIME writers SHOULD copy Bcc recipients to the TNEF **recipient table** for attached messages.

### 2.1.1.2 Reply-to

To generate a Reply-To MIME header, clients set the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties, as specified in [MS-OXOMSG].

When generating MIME, MIME writers generate a Reply-To MIME header by using the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties. MIME writers SHOULD ignore the PidTagReplyRecipientNames value if the COUNT of names does not match the COUNT of entries in the PidTagReplyRecipientEntries property. Assuming the counts do match, each entry in the value of the PidTagReplyRecipientNames property maps to one display name, and each EntryID in the value of the PidTagReplyRecipientEntries property maps to one address, as follows:

1. If bytes 4-19 are equal to the MUIDEMSAB UUID value "{%xdc.a7.40.c8.c0.42.10.1a.b4.b9.08.00.2b.2f.e1.82}", it is an address book EntryID. In this case, the MIME writer SHOULD look up the address book entry that corresponds to the DN that is contained in the EntryID, and use its primary SMTP proxy address.

2. If bytes 4-19 are equal to the MUIDOOP UUID value "{%x81.2b.1f.a4.be.a3.10.19.9d.6e.00.dd.01.0f.54.02}", it is a one-off EntryID. The e-mail type and address are encoded in the one-off EntryID ([MS-OXCDATA] section 2.2.5.1) and SHOULD be extracted. If the e-mail type is SMTP, the e-mail address SHOULD be used as is; otherwise, the address MUST be IMCEA -encapsulated, as specified in section 2.1.1.8.

When generating TNEF, MIME writers SHOULD also copy the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties to the attMsgProps attribute in the TNEF body part, as specified in [MS-OXTNEF].<8>

#### 2.1.1.3 From

To generate a From MIME header, clients set the **PidTagSentRepresenting** property group.

When generating MIME, MIME writers generate a From header by using the values of the **PidTagSentRepresenting** property group. The order of preference in that property group is as specified in section 2.1.1.1.

When generating TNEF, MIME writers SHOULD also copy the values of the **PidTagSentRepresenting** property group to **attSentFor** and **attMsgProps** in the TNEF body part, as specified in [MS-OXTNEF].<9>

#### 2.1.1.4 Sender

To generate a Sender MIME header, clients set the value of the PidTagSender property group.

MIME writers generate a Sender header by using the values of the PidTagSender property group, which consists of all the properties containing the "PidTagSender" prefix. The order of preference in that property group is as specified in section 2.1.1.1. MIME writers SHOULD NOT generate the Sender header if the PidTagSender property group and the **PidTagSentRepresenting** property group represent the same user.<10>

When generating TNEF, MIME writers SHOULD also copy the values of the PidTagSender property group to attFrom and attMsgProps in the TNEF body part, as specified in [MS-OXTNEF].<11>

#### 2.1.1.5 Return Receipt

A MIME writer generates a (nonstandard) Return-Receipt-To header when the PidTagOriginatorDeliveryReportRequested property is set to TRUE.

If the PidTagOriginatorDeliveryReportRequested property is set and its value is TRUE, MIME writers MUST copy the Return-Receipt-To header from one of the following property groups in the specified order of priority and according to the method specified in section 2.1.1.1:<12>

- **PidTagReadReceipt** property group
- **PidTagSender** property group
- **PidTagSentRepresenting** property group

When generating TNEF, MIME writers SHOULD copy the values of the PidTagOriginatorDeliveryReportRequested and one of the e-mail address properties specified above to attMsgProps in the TNEF body part, as specified in [MS-OXTNEF] <13>.

### 2.1.1.6 Read Receipt

To generate a Disposition-Notification-To MIME header, protocol clients set the PidTagReadReceiptRequested property to TRUE and also set the values of either the **PidTagReadReceipt** property group or the **PidTagSentRepresenting** property group. The **PidTagReadReceipt** property group consists of all the properties whose names contain the "PidTagReadReceipt" prefix. The **PidTagSentRepresenting** property group consists of all the properties whose names contain the "PidTagSentRepresenting" prefix.

MIME writers check the PidTagReadReceiptRequested property value first. If the property is not set or the value is FALSE, MIME writers MUST NOT generate the Disposition-Notification-To header.

If the PidTagReadReceiptRequested property is set and its value is TRUE, MIME writers generate the Disposition-Notification-To header from the **PidTagReadReceipt** property group, if that property group is set. The order of preference in that property group is as specified in section 2.1.1.1. If the **PidTagReadReceipt** property group is not set, protocol servers SHOULD generate the Disposition-Notification-To header from the **PidTagSentRepresenting** property group.

MIME writers MUST generate the Disposition-Notification-To MIME header as specified in [RFC3798].

When generating TNEF, MIME writers SHOULD also copy the values of the PidTagReadReceiptRequested, the **PidTagReadReceipt**, and the **PidTagSentRepresenting** groups of properties to attMsgProps in the TNEF body part, as specified in [MS-OXTNEF].<14>

### 2.1.1.7 Directory Lookups

Clients SHOULD specify the primary SMTP proxy address or the address book (EX) proxy address in all address elements of a message. But clients can use any proxy address. MIME writers SHOULD perform a lookup on the proxy address in the address book directory, as specified in [MS-OXOABK]. If a matching address book entry is found, a MIME writer SHOULD substitute its primary SMTP proxy address for the address specified by the client.

### 2.1.1.8 IMCEA Encapsulation

When no SMTP proxy address is available for an address element, protocol servers SHOULD encapsulate any other address type to produce the required SMTP address, using the Internet Mail Connector Encapsulated Address (IMCEA) encapsulation mechanism.<15> The **domain** part of the encapsulated SMTP address SHOULD be the MIME writer's local domain or the domain of another mail server that can de-encapsulate, and deliver to, the encapsulated address.

The IMCEA encapsulation mechanism is defined for the address types listed in the following table.

| Address Type | Value of PidTagAddressType or related property |
|---|---|
| Address Book | "EX" |
| Facsimile | "FAX" |
| X.400 | "X400" |

```
Encapsulated-address = "IMCEA" address-type "-" encoded-address "@" domain
address-type = *VCHAR
```

```
domain = dot-atom-text; see [RFC2822] section 3.2.4 for the definition.

encoded-address = * (Escaped-chars/ Normal-chars)

Escaped-chars = (ESCSLASH / ESCCHARS)

; Encoded form for "/" (%x2F) is "_"
ESCSLASH = %x5F

; All OCTETS not ALPHA, DIGIT, or in "-=/"
; These are a "+" and the two hex digits of the OCTET's value.
ESCCHARS = "+" 2(HEXDIG)

; All other characters
Normal-chars= (ALPHA / DIGIT / HYPHEN / EQUALSIGN)
HYPHEN = %x2D
EQUALSIGN = %x3D
```

Encapsulated addresses MUST NOT include line breaks, and therefore can require longer line lengths than those recommended by [RFC2822] <16><17>.

#### 2.1.1.9 PidTagAddressType

The value of PidTagAddressType is a string that names the messaging system that the address is destined for. It is used to assign responsibility for an e-mail address to the right transport provider. The string value provided by PidTagAddressType contains only uppercase alphabetic characters from "A" through "Z", and the numbers from "0" through "9". The value of PidTagAddressType is also used to designate the correct format for the e-mail address itself, PidTagEmailAddress.

If a client tries to compose a message to a user whose address type is not in the server's list of known address types, the message will produce a **non-delivery report (NDR)** unless the client itself, acting as the message transfer agent, is able to deliver the message by using an alternate transport that bypasses the server.

The following table lists the address types that are known at this time. The common address types include "EX", "SMTP", "X400", and "X500".

| Messaging system | PidTagAddressType value |
|---|---|
| Microsoft® Exchange Server | "EX" |
| Internet | "SMTP" |
| X.400 Message Handling System | "X400" |
| X.500 Directory Services | "X500" |

### 2.1.2 Envelope Elements

Many message object properties that map to MIME headers have string values. Unless otherwise specified, the string values are simply copied from the property to the header. When MIME writers generate MIME header values, the encoding specified in [RFC2047] MUST be used to encode **Unicode** characters according to the conditions specified in section 1 of that document.

Likewise, unless otherwise specified, when a MIME message with a TNEF body part is being generated, all message object properties SHOULD be copied to the **attMsgProps** attribute of the TNEF body part, even if there is also a corresponding MIME header.<18>

#### 2.1.2.1 Message Class

When generating TNEF, MIME writers copy the value of the PidTagMessageClass property to **attMsgProps** in the TNEF body part, as specified in [MS-OXTNEF]. In addition, MIME writers SHOULD map the value of the PidTagMessageClass property to the **attMessageClass** attribute, as specified in [MS-OXTNEF].

When generating pure MIME, the value of PidTagMessageClass SHOULD NOT be copied to MIME messages. Instead, its value is reflected in the structure of the MIME message, as specified in the following table. The MIME structure is indicated by listing the value of the Content-Type header, indented according to how the MIME entities are nested.

| PidTagMessageClass value | MIME structure |
|---|---|
| "IPM.Note.SMIME.MultipartSigned", or begins with "IPM.InfoPathForm." and ends with ".SMIME.MultipartSigned" | Multipart/signed, as specified in [RFC5751] and [MS-OXOSMIME]. |
| "IPM.Note.SMIME", or begins with "IPM.InfoPathForm." and ends with ".SMIME" | Application/pkcs7-MIME, as specified in [RFC5751] and [MS-OXOSMIME]. |
| "REPORT.IPM.Note.DR" or "REPORT.IPM.Note.NDR" (other values MAY be substituted for "IPM.Note") | As specified in [RFC3464]:<br>multipart/report<br>text/HTML<br>message/delivery-status<br><original message structure> |
| "REPORT.IPM.Note.IPNRN" or "REPORT.IPM.Note.IPNNRN" (other values MAY be substituted for "IPM.Note") | As specified in [RFC3798]:<br>multipart/report<br>text/HTML<br>Message/disposition-notification |
| Equals "IPM.Appointment" or begins with "IPM.Appointment." | Text/**calendar**, as specified in [RFC2445] and [MS-OXCICAL].<19> |
| Begins with "IPM.Schedule.Meeting." | Content mapped to text/calendar, as specified in [RFC2445] and [MS-OXCICAL].<br>**Top-level message** structure is multipart/alternative or multipart/mixed, depending on the presence and type of message body and attachments. For details, see section 2.1.3. |
| "IPM.Note" or any other value | Text/plain, text/HTML, multipart/alternative, multipart/related (specified in [RFC2387]), or multipart/mixed, depending on the presence and type of message body and attachments. For details, see section 2.1.3. |

### 2.1.2.2 Content Class

MIME writers SHOULD generate the following values for a Content-class header, based on the value of the PidTagMessageClass property.<20>

| PidTagMessageClass value | Content-class header value |
|---|---|
| "IPM.Note.Microsoft.Fax" | "fax" |
| "IPM.Note.Microsoft.Fax.CA" | "fax-ca"<21> |
| "IPM.Note.Microsoft.Missed.Voice" | "missedcall" |
| "IPM.Note.Microsoft. Conversation.Voice" | "voice-uc" |
| "IPM.Note.Microsoft.Voicemail.UM.CA" | "voice-ca"<22> |
| "IPM.Note.Microsoft.Voicemail.UM" | "voice" |

| PidTagMessageClass prefix | Content-class header value |
|---|---|
| "IPM.Note.Custom." | "urn:content-class:custom.", followed by the value of the PidTagMessageClass property with the "IPM.Note.Custom." prefix removed. |
| "IPM.InfoPathForm." | If the PidLidInfoPathFormName property has some value, the Content-class header SHOULD be generated with the value of "InfoPathForm.", followed by a string, which is generated as follows:<br>MIME writers SHOULD take the value of the PidTagMessageClass property, and remove the "IPM.InfoPathForm." prefix.<br>If the remaining string contains a '.' symbol, the value SHOULD be truncated before the period '.'.<br>The value of the PidLidInfoPathFormName property SHOULD be appended to this string, preceded by a '.' character. |

If the MIME writer was unable to find a mapping between the value of the PidTagMessageClass property and a value of the Content-class MIME header, it SHOULD look up the value of the PidNameContentClass property. If this property has a value, that value SHOULD be used as the value of the Content-class header; otherwise, a header SHOULD NOT be generated.

### 2.1.2.3 Unified Messaging Properties

To generate an X-CallingTelephoneNumber header,<23> clients SHOULD set the value of the PidTagSenderTelephoneNumber property. They MAY instead use PidNameXSenderTelephoneNumber. MIME writers SHOULD copy either property to the X-CallingTelephoneNumber header, preferring PidTagSenderTelephoneNumber.<24><25>

To generate an X-VoiceMessageDuration header, clients SHOULD set the value of the PidTagVoiceMessageDuration property. They MAY instead use PidNameXVoiceMessageDuration. MIME writers SHOULD map either property to the X-VoiceMessageDuration header, preferring PidTagVoiceMessageDuration.<26><27> The value of the PidTagVoiceMessageDuration property is a positive valued PtypInteger32 and is formatted as a decimal string in the header without sign or separator characters.

To generate an X-VoiceMessageSenderName header, clients SHOULD set the value of the PidTagVoiceMessageSenderName property. They MAY instead use PidNameXVoiceMessageSenderName. MIME writers SHOULD copy either property to the X-VoiceMessageSenderName header, preferring PidTagVoiceMessageSenderName.<28><29>

To generate an X-FaxNumberOfPages header, clients SHOULD set the value of the PidTagFaxNumberOfPages property. They MAY instead use PidNameXFaxNumberOfPages. MIME writers SHOULD map either property to the X-FaxNumberOfPages header, preferring PidTagFaxNumberOfPages.<30><31> The value of the PidTagFaxNumberOfPages property is a positive valued PtypInteger32 and is formatted as a decimal string in the header without sign or separator characters.

To generate an X-AttachmentOrder header, clients SHOULD set the value of the PidTagVoiceMessageAttachmentOrder property. They MAY instead use PidNameXVoiceMessageAttachmentOrder. MIME writers SHOULD copy either property to the X-AttachmentOrder header, preferring PidTagVoiceMessageAttachmentOrder.<32><33>

To generate an X-CallID header, clients SHOULD set the value of the PidTagCallId property. They MAY instead use PidNameXCallId. MIME writers SHOULD copy either property to the X-CallID header, preferring PidTagCallId.<34><35>

#### 2.1.2.4 Arbitrary MIME Headers

To generate an arbitrary header on a MIME message, a client creates a named property in the **PS_INTERNET_HEADERS property set**, with the **property name** equal to the header name and the data type equal to string. The value of this property is set to the MIME header value.

MIME writers use the name and value of such a property to create a header on the generated MIME message with the corresponding name and value. MIME writers encode the header value as specified in [RFC2047], according to the conditions specified in section 1 of that document.<36> But MIME writers MUST NOT create such a header if a different Message object property is already mapped to the same header, or if the header name begins with one of the reserved name prefixes "X-Microsoft-Exchange-Organization" or "X-Microsoft-Exchange-Forest".<37>

#### 2.1.2.5 Importance

To generate an Importance header, a client sets the value of the PidTagImportance property as specified in the following table.

| PidTagImportance value | Importance header value |
|---|---|
| 0x00000000 | Low |
| 0x00000001 | Normal |
| 0x00000002 | High |

MIME writers MUST map the value of the PidTagImportance property to the Importance header, as specified in the table. MIME writers MAY omit the Importance header for a PidTagImportance value of 1 (normal) or for values other than 0, 1, or 2.

When generating TNEF, MIME writers also copy the value of the PidTagImportance property to the attPriority and attMsgProps attributes, as specified in [MS-OXTNEF].

### 2.1.2.6 Sensitivity

To generate a Sensitivity header, a client sets the value of the PidTagSensitivity property as specified in the following table.

| PidTagSensitivity value | Sensitivity header value |
|---|---|
| 0x00000000 | Normal |
| 0x00000001 | Personal |
| 0x00000002 | Private |
| 0x00000003 | Company-Confidential |

MIME writers MUST map the value of the PidTagSensitivity property to the Sensitivity header, as specified in the table. MIME writers MAY omit the Sensitivity header value for a PidTagSensitivity value of 0 (normal) or for values other than 0, 1, 2, or 3.

When generating TNEF, MIME writers also copy the value of the PidTagSensitivity property to the attMsgProps attribute, as specified in [MS-OXTNEF].

### 2.1.2.7 Sent Time

To generate a Date header, clients set the value of the PidTagClientSubmitTime property. The property value is expressed in **Coordinated Universal Time (UTC)**.

MIME writers copy the value of the PidTagClientSubmitTime property to the Date header, formatting it as specified by [RFC2822] section 3.3. MIME writers SHOULD include hours, minutes, and seconds in the generated Date header value. MIME writers MAY convert the date and time value from UTC to another time zone of their choice.

If no value is specified for PidTagClientSubmitTime when a message is submitted to SMTP, MIME writers SHOULD generate a Date header with a value of the current time.

When generating TNEF, MIME writers SHOULD also copy the value of the PidTagClientSubmitTime property to the attDateSent and attMsgProps attributes, as specified in [MS-OXTNEF].<38>

### 2.1.2.8 Subject

To generate a Subject header, clients SHOULD set the PidTagSubjectPrefix and PidTagNormalizedSubject properties on the message object. Clients MAY set the PidTagSubject property instead, but in that case, the separation of subject from subject prefix is vulnerable to limitations of the server's parsing procedure, which is specified in section 2.2.2.6.1. Subject property values SHOULD NOT contain line breaks.

MIME writers SHOULD generate the Subject header by combining the values of the PidTagSubjectPrefix and PidTagNormalizedSubject properties.<39>

If those two properties are not available, MIME writers MUST copy the value of the PidTagSubject property to the Subject header. MIME writers MAY truncate the subject value. The property value SHOULD NOT be truncated in the middle of a multibyte character.

When generating TNEF, MIME writers SHOULD also copy the message subject (however it is obtained) to the attSubject and attMsgProps attributes, as specified in [MS-OXTNEF].<40> MIME

writers SHOULD also copy the PidTagSubjectPrefix and PidTagNormalizedSubject properties, with their values, to the **attMsgProps** attribute.

#### 2.1.2.9 Conversation Topic

To generate a Thread-Topic header, clients set the value of the PidTagConversationTopic property. Clients SHOULD set this property to the same value as PidTagNormalizedSubject, with any subject prefix removed, as specified in section 2.2.2.6.1.

MIME writers copy the value of the PidTagConversationTopic property to the Thread-Topic header.

#### 2.1.2.10 Conversation Index

To generate a Thread-Index header, clients set the value of the PidTagConversationIndex property, as specified in [MS-OXOMSG].

MIME writers copy the value of the PidTagConversationIndex property to the Thread-Index header. The **property type** is binary; the value is encoded using **base64 encoding**, as specified in [RFC2045].

#### 2.1.2.11 Message ID

MIME writers SHOULD copy the value of the PidTagInternetMessageId property to the Message-ID header.<41> If no value is specified for PidTagInternetMessageId when a message is submitted to SMTP, MIME writers SHOULD generate a value as specified in [RFC2822].

Clients SHOULD NOT set the PidTagInternetMessageId property when submitting a message via **remote procedure call (RPC)**. As specified in [RFC2822], the value of message-ID is unique, and is assigned by the server that originated the message. Servers MAY overwrite PidTagInternetMessageId from a client before submitting the message to SMTP.

Once set, the value of the message-ID header and the corresponding property value, PidTagInternetMessageId, SHOULD remain constant. MIME writers SHOULD NOT overwrite the value of PidTagInternetMessageId when generating MIME for protocols such as POP/IMAP.

#### 2.1.2.12 References

To generate a References header, clients set the value of the PidTagInternetReferences property.

MIME writers copy the value of the PidTagInternetReferences property to the References header.

#### 2.1.2.13 Categories

To generate a Keywords header, clients set the value of the PidNameKeywords property. The type of PidNameKeywords is multiple strings; each category SHOULD be mapped to a single keyword.

MIME writers SHOULD copy each sub-value of the PidNameKeywords property to a separate keyword in the Keywords header, with a comma (U+002C) and space (U+0020) separating each keyword. MIME writers can drop the PidNameKeywords instead of copying it to the Keywords header, to avoid conflict among different sets of categories in different organizations.

#### 2.1.2.14 In-Reply-To Message ID

To generate an In-Reply-To header, clients set the value of the PidTagInReplyToId property.

MIME writers copy the value of the PidTagInReplyToId property to the In-Reply-To header.<42>

### 2.1.2.15 List Server Properties

To generate a List-Help header, clients set the value of the PidTagListHelp property.

MIME writers copy the value of the PidTagListHelp property to the List-Help header.

To generate a List-Subscribe header, clients set the value of the PidTagListSubscribe property.

MIME writers copy the value of the PidTagListSubscribe property to the List-Subscribe header.

To generate a List-Unsubscribe header, clients set the value of the PidTagListUnsubscribe property.

MIME writers copy the value of the PidTagListUnsubscribe property to the List-Unsubscribe header.

The List-Help, List-Subscribe, and List-Unsubscribe headers are specified in [RFC2369].

### 2.1.2.16 Language Properties

To generate an [RFC3282] Accept-Language header, clients set the value of the PidNameAcceptLanguage property.

MIME writers SHOULD copy the value of the PidNameAcceptLanguage property to the Accept-Language header.<43><44> If the PidNameAcceptLanguage property is missing, MIME writers SHOULD identify the acceptable locales of the sender's **mailbox** and write the corresponding language tag, as specified by [RFC4646], as the value of the Accept-Language header.

To generate an [RFC3282] Content-Language header, clients set the value of the PidTagMessageLocaleId property to a locale ID.<45>

MIME writers use the value of the PidTagMessageLocaleId property to write the Content-Language header. The value of PidTagMessageLocaleId is an LCID (a 32-bit integer value), but the header value is a language tag, as specified by [RFC4646]. Mapping between LCID and language tag is done as specified in [MS-LCID].

### 2.1.2.17 Classification Properties

To generate headers related to message classification, clients both set the value of the PidLidClassified property to TRUE and set the values of the following properties: PidLidClassification, PidLidClassificationDescription, PidLidClassificationGuid, and PidLidClassificationKeep.

When the value of the PidLidClassified property is TRUE, MIME writers SHOULD copy all classification property values to their corresponding headers, as specified in the following table.<46><47> If the value of PidLidClassified is FALSE, then no value is written for any of the five headers listed in the following table.

| Classification property | Classification header | Property value mapping |
|---|---|---|
| PidLidClassified | X-Microsoft-Classified | TRUE maps to "true".<br>If the value of PidLidClassified is FALSE, then there is no header. |
| PidLidClassificationKeep | X-Microsoft-ClassKeep | TRUE maps to "true".<br>FALSE maps to no header. |
| PidLidClassification | X-Microsoft- | No mapping. The string value is copied directly. |

| Classification property | Classification header | Property value mapping |
|---|---|---|
| | Classification | |
| PidLidClassificationDescription | X-Microsoft-ClassDesc | No mapping. The string value is copied directly. |
| PidLidClassificationGuid | X-Microsoft-ClassID | No mapping. The string value is copied directly. |

#### 2.1.2.18 Payload Properties

To generate an X-Payload-Provider-GUID header, clients set the value of the PidTagAttachPayloadProviderGuidString property.<48>

MIME writers SHOULD copy the value of the PidTagAttachPayloadProviderGuidString property to the X-Payload-Provider-GUID header.<49><50><51>

To generate an X-Payload-Class header, clients set the value of the PidTagAttachPayloadClass property.

MIME writers SHOULD copy the value of the PidTagAttachPayloadClass property to the X-Payload-Class header.<52><53><54>

#### 2.1.2.19 Has Attach

To generate an X-MS-Has-Attach header, clients MUST add at least one attachment to the **attachments table** of the message object.

When the message object's attachments table contains at least one attachment, MIME writers SHOULD generate an X-MS-Has-Attach header with a value of "Yes".<55>When the message object's attachments table is empty, MIME writers generate an X-MS-Has-Attach header with no value.

#### 2.1.2.20 Auto Response Suppress

To generate an X-Auto-Response-Suppress header, clients set the value of the PidTagAutoResponseSuppress property.

When the PidTagAutoResponseSuppress property has a value of 0 (zero) or -1, MIME writers SHOULD map its value to the X-Auto-Response-Suppress header as shown in the following table.<56><57>

| PidTagAutoResponseSuppress property value | X-Auto-Response-Suppress header value |
|---|---|
| 0 | "None" |
| -1 | "All" |

When the PidTagAutoResponseSuppress property has a value other than 0 (zero) or -1, MIME writers MUST construct the value of the X-Auto-Response-Suppress header as follows: For each bit of the value of PidTagAutoResponseSuppress that is set (left-hand column), append the string in the center column to the header value. If the header value was nonempty, append a comma (U+002C) and space (U+0020) before the new value.

| PidTagAutoResponseSuppress property value | X-Auto-Response-Suppress header value | Description |
|---|---|---|
| 0x00000001 | "DR" | Suppress delivery reports from transport. |
| 0x00000002 | "NDR" | Suppress non-delivery reports from transport. |
| 0x00000004 | "RN" | Suppress read notifications from receiving client. |
| 0x00000008 | "NRN" | Suppress non-read notifications from receiving client. |
| 0x00000010 | "OOF" | Suppress **Out of Office (OOF)** notifications. |
| 0x00000020 | "AutoReply" | Suppress auto-reply messages other than OOF notifications. |

For example, if the value of PidTagAutoResponseSuppress is 0x000C, the header MUST be written as:

```
X-Auto-Response-Suppress: RN, NRN
```

(**Note:** The order of these values in the header is not important.)

#### 2.1.2.21 Is Auto Forwarded

To generate an X-MS-Exchange-Organization-AutoForwarded header, clients set the value of the PidTagAutoForwarded property to TRUE.

If the value of the PidTagAutoForwarded property is TRUE, MIME writers SHOULD generate the following header:<58><59>

```
X-MS-Exchange-Organization-AutoForwarded: true
```

If the property is absent or the property value is false, a header SHOULD NOT be generated.

#### 2.1.2.22 Sender Id Status

To generate an X-MS-Exchange-Organization-SenderIdResult header, clients set the value of the PidTagSenderIdStatus property.

MIME writers SHOULD copy the value of the PidTagSenderIdStatus property, which is a PtypInteger32, to the X-MS-Exchange-Organization-SenderIdResult header, formatting it as a string, without separator characters.<60><61>

#### 2.1.2.23 Purported Sender Domain

To generate an X-MS-Exchange-Organization-PRD header, clients set the value of the PidTagPurportedSenderDomain property.

MIME writers SHOULD copy the value of the PidTagPurportedSenderDomain property to the X-MS-Exchange-Organization-PRD header.<62><63>

#### 2.1.2.24 Spam Confidence Level

To generate an X-MS-Exchange-Organization-SCL header, MIME writers set the value of the PidTagContentFilterSpamConfidenceLevel property to a value in the range "-1" to "10". The value of "-1" indicates the message is from a trusted sender and is never treated as **spam**. Values "0" through "10" indicate the confidence levels calculated from the message content, as specified in [MS-OXCSPAM].

MIME writers SHOULD copy the value of the PidTagContentFilterSpamConfidenceLevel property, which is a PtypInteger32, to the X-MS-Exchange-Organization-SCL header, formatting it as a decimal numeric string without separator characters.<64><65>

#### 2.1.2.25 Flag Request

To generate an X-Message-Flag header, clients set the value of the PidLidFlagRequest property.

MIME writers copy the value of the PidLidFlagRequest property to the X-Message-Flag header.

#### 2.1.2.26 TNEF Correlation Key

When creating a new TNEF message, MIME writers choose a unique key relating the TNEF body part to its parent message. (MIME writers SHOULD use the value of PidTagInternetMessageId for this purpose.) The chosen value MUST be written in two places:

As the value of the X-MS-TNEF-Correlator header on the MIME message.

As the value of PidTagTnefCorrelationKey in the **attMsgProps** attribute of the TNEF body part itself.

This pair of values SHOULD be used by MIME writers to validate that the **top-level message** and its TNEF body part do, in fact, belong to each other, and are not (for example) the result of a non-TNEF-aware **Mail User Agent (MUA)** forwarding a message with an attached TNEF body part and retaining the attachment.

#### 2.1.2.27 Received Headers

If a MIME message is bound for SMTP, MIME writers MUST NOT copy Received headers from PidTagTransportMessageHeaders to the generated MIME header. If a MIME message is bound for POP3 or IMAP4, MIME writers SHOULD copy all Received headers from PidTagTransportMessageHeaders to the generated MIME header.

#### 2.1.2.28 ReplyBy Time

To generate a Reply-By header, clients set the PidTagReplyTime property to the date and time by which a reply is requested. Clients also MUST set the PidLidFlagRequest property, as specified in section 2.1.2.25, to any non-empty string value. If PidLidFlagRequest is not set, the Reply-By header will not be generated.

MIME writers copy the value of the PidTagReplyTime property to the Reply-By header.<66>

#### 2.1.2.29 Content-ID

To generate the Content-ID header, clients set the value of the PidTagBodyContentId property.

#### 2.1.2.30 XRef

A MIME writer generates an XRef header when it detects a named custom property in the PS_INTERNET_HEADERS property with a property name of "XRef". The value of this property becomes the header value. Setting a value in the PS_INTERNET_HEADERS property set is specified in section 2.2.2.29.

The XRef header maps to the PidNameCrossReference property, as specified in section 2.2.2.27.

### 2.1.3 Body Text

When generating pure MIME, MIME writers generate a single MIME entity for the message body, and it MUST be the first entity generated. (For message objects without attachments, it SHOULD be the only MIME entity generated.) The MIME entity generated for the message body can have several different structures, some of them fairly complex.

For diagrams of message structure with attachments, and for details about how to determine whether an **Attachment object** represents an inline attached file, see section 2.1.4.

#### 2.1.3.1 Client Actions

To create a plain text message body in MIME, clients SHOULD set the value of the PidTagBody property. Additionally, clients SHOULD set the value of the PidTagInternetCodepage property to a **code page** that corresponds to the charset that the client wants to appear in MIME. If the client does not set PidTagInternetCodepage for a plain text message body, the server SHOULD<67> default this property to ISO-8859-15. Clients SHOULD NOT create inline Attachment objects when the **best body** format of the message object is plain text.

To create an HTML message body in MIME, clients SHOULD set the value of the PidTagHtml property to the HTML message text. When this property is set, clients MUST set the value of the PidTagInternetCodepage to the code page of the HTML message text. Note that PidTagHtml is a **PtypBinary** property, not a **PtypString** property. Clients can, instead, set the value of the PidTagRtfCompressed property to the body text in compressed RTF format, depending on the MIME writer that is used to convert this text to HTML format. Clients MUST NOT create HTML message text in Unicode (**UTF-16LE**), and the value of PidTagInternetCodepage MUST NOT be set to "1200". UTF-32 and UTF-16GE are also not acceptable for this purpose; UTF-7 (code page 65000) and UTF-8 (code page 65001) are acceptable.

To create a multipart/related message body in MIME with HTML body text and inline images, clients SHOULD set the value of the PidTagHtml property to the HTML message text. When this property is set, the value of the PidTagInternetCodepage property is set to the code page of the HTML message text. (Note that PidTagHtml is a **PtypBinary** property, not a **PtypString** property.) Clients supply a value for either the PidTagAttachContentId or the PidTagAttachContentLocation property on related file attachments such as images; PidTagAttachContentId SHOULD be chosen for this purpose. Depending on the choice of attachment property, inline image links in the HTML body MUST use one of the following:

The "cid:" **Uniform Resource Identifier (URI)** scheme and a unique content identifier that matches the value of the PidTagAttachContentId property on the corresponding Attachment object.

A copy of the value of the PidTagAttachContentLocation property (see [MS-OXCMSG]) on the corresponding Attachment object.

Instead of setting the value of the PidTagHtml property, clients can set the value of the PidTagRtfCompressed property and include OLE attachments, depending on the protocol server to

convert the RTF text to HTML and the static renderings of the OLE attachments to image attachments. For details, see section 2.1.3.7.

For plain text messages, clients SHOULD write the value of the PidTagBody property in Unicode and SHOULD set the value of the PidTagInternetCodepage property to the code page that matches the sender's preferred charset. When generating a MIME element for the plain text body, MIME writers map this code page to a charset name, convert the Unicode text into that charset, and write that charset name to the value of the charset parameter of the Content-Type header. The plain text MIME element generated for a TNEF message SHOULD be treated in the same way.

For HTML messages, clients SHOULD write the value of the PidTagHtml property by using text in the sender's preferred charset. Clients set the value of the PidTagInternetCodepage property to the code page that corresponds to the preferred charset. Clients MUST NOT use UTF-16 (code page 1200) as the preferred charset. If the HTML document contains a content-type meta tag, its charset parameter value SHOULD match the preferred charset.

When generating a MIME element or elements for an HTML message body, MIME writers map the value of the PidTagInternetCodepage property to a charset name, write the MIME element body in that charset, and write that charset name as the value of the Content-Type **header's** charset parameter. If the HTML document contains a content-type meta tag, its charset parameter value SHOULD match the Content-Type header's charset parameter value.

For RTF messages, clients SHOULD write the value of the PidTagRtfCompressed property by using text in the sender's preferred charset. Clients SHOULD set the value of the PidTagInternetCodepage property to the code page that corresponds to the preferred charset. The preferred charset MUST NOT be UTF-16 (code page 1200). MIME writers MUST NOT rely on the value of the PidTagInternetCodepage property, but treat it as a preference; MIME writers SHOULD instead rely on the value of one or more \ansicpg elements in the RTF stream, as specified in [MSFT-RTF], to determine the actual body code page.

When generating a MIME element or elements for an RTF message body, MIME writers SHOULD convert the RTF text to plain text or HTML, SHOULD map the body code page to a charset name, SHOULD write the MIME element body in that charset, and SHOULD write that charset name as the value of the Content-Type header's charset parameter.<68>

Even if a message object has no body, clients SHOULD set the value of the PidTagInternetCodepage property to indicate a preferred charset for header text, to be used in [RFC2047] encoding.

When generating headers for a MIME entity, it can be necessary to encode the characters as specified in [RFC2047]. MIME writers SHOULD use the same charset for all headers and the message body.<69>Attachments that are themselves messages are independent and can have a different charset.

### 2.1.3.2 Message Body in TNEF

When generating TNEF, MIME writers SHOULD identify the "best body" property of the Message object, as specified in [MS-OXBBODY], and copy its value to the **attMsgProps** attribute of the TNEF body part. MIME writers also place a plain text version of the message body in the first child body part of the TNEF message, as specified in section 2, generating plain text from the value of the "best body" property if the best body format is other than plain text. Finally, when the best body is plain text, MIME writers SHOULD also write a matching value for the PidTagRtfCompressed property to the **attMsgProps** attribute of the TNEF body part. <70>

### 2.1.3.3 Simple Plain Text Message Body

When the best body format type is plain text, MIME writers SHOULD generate a single MIME entity with the value of its Content-Type header set to text/plain.

The charset parameter value of this MIME entity's Content-Type header SHOULD be set to a charset that corresponds to the value of the PidTagInternetCodepage property (a code page number). If there is no PidTagInternetCodepage property, the value of the PidTagMessageCodepage property can be used instead, but in that case the message code page SHOULD first be mapped to the corresponding Internet code page. MIME writers SHOULD verify that the plain text, which is stored as UTF-16, can actually be encoded in this charset and SHOULD, if necessary, choose a different charset that can in fact encode the entire message body; the code page properties express a preference rather than a requirement.

The value of the PidTagBody property is written to the content of the text/plain MIME element, after being converted to the chosen charset.

### 2.1.3.4 HTML Text Message Body Without Inline Attachments

When the best body format type is HTML and no inline Attachment objects exist, MIME writers SHOULD generate a MIME entity with multipart/alternative for the value of its Content-Type header, and with the following two child entities:

1. The first child **entity** has "text/plain" for the value of its Content-Type header. Its body SHOULD be plain text generated from the value of the PidTagHtml property. The body MAY instead be copied from the value of the PidTagBody property, assuming that PidTagBody is equal to PidTagHtml<71>.

2. The second child entity has "text/HTML" for the value of its Content-Type header. Its body is the value of the PidTagHtml property.

The plain text and charset parameters SHOULD be processed as specified in section 2.1.3.3. HTML text MAY be processed in exactly the same way, or characters that do not fit the preferred charset can instead be encoded within the HTML.

### 2.1.3.5 HTML Text Message Body from RTF Without Inline Attachments

When the best body format type is RTF and no inline (OLE) Attachment objects exist, MIME writers SHOULD generate a multipart/alternative MIME entity with the following two child entities:

1. The first child entity has "text/plain" for the value of its Content-Type header. Its body SHOULD be plain text generated from the value of the PidTagRtfCompressed property, but can instead be copied from the value of the PidTagBody property, assuming that it contains substantially similar text.

2. The second child entity has "text/HTML" for the value of its Content-Type header. Its body is HTML text. The HTML text SHOULD be generated from the value of the PidTagRtfCompressed property, but can instead be copied form the value of the PidTagHtml property, assuming that it contains substantially similar text.

The text and charset parameters SHOULD be processed as specified in section 2.1.3.4.

#### 2.1.3.6 HTML Text Message Body with Inline Attachments

When the best body format type is HTML and inline Attachment objects exist, MIME writer SHOULD generate a MIME entity with "multipart/related" for the value of its Content-Type header and two or more child elements, as follows:

1. The first child entity is a "multipart/alternative" structure, as specified in section 2.1.3.4

2. Subsequent child entities are generated from the message object's inline attachments. A child entity MUST be generated if and only if the Attachment object is marked as specified in section 2.1.4.1.

   MIME writers SHOULD verify that the HTML text actually contains a reference to each inline attachment object, either by its PidTagAttachContentId or PidTagAttachContentLocation property, as specified in 2.1.3.1. If the HTML text contains no such reference, the MIME writer SHOULD consider this Attachment object as not inline and generate its MIME entity as a peer of the multipart/related MIME entity, instead of as its child.<72>

#### 2.1.3.7 HTML Text Message Body from RTF with Inline (OLE) Attachments

When the best body format type is RTF and inline (OLE) Attachment objects exist, MIME writers SHOULD generate a MIME entity with multipart/related for the value of its Content-Type header and three or more child entities, as follows:

1. The first child entity is a multipart/alternative structure, as specified in section 2.1.3.5.

2. Subsequent child entities are generated from the Message object's inline attachments. Each entity is generated as specified in 2.1.4.4, because inline Attachment objects in RTF messages are always OLE attachments.

#### 2.1.3.8 Calendar Items and Meeting Messages

A message object is a calendar item when the value of PidTagMessageClass starts with "IPM.Appointment." or equals "IPM.Appointment". A message object is a meeting message when the value of PidTagMessageClass starts with "IPM.Schedule.Meeting.". Clients SHOULD create items of these types with a best body format type of RTF. Clients can use a plain text body instead, but SHOULD NOT create calendar items or meeting messages with a best body format type of HTML.

Each of the leaf MIME entities specified in this section SHOULD use UTF-8 as its charset, as specified in [RFC2445].<73>

##### 2.1.3.8.1 Plain Text Calendar Message

When the best body format type of a calendar item or meeting message is plain text, MIME writers SHOULD generate a MIME entity with multipart/alternative for the value of its Content-Type header and two child entities, as follows:

1. The first child entity has "text/plain" for the value of its Content-Type header, and its content is copied from PidTagBody.

2. The second child entity has text/calendar for the value of its Content-Type header, and its content is generated as specified in [MS-OXCICAL].

#### 2.1.3.8.2 Calendar Message Without Inline Attachments

When the best body format type of a calendar item or meeting message is RTF and there are no inline attachments, MIME writers SHOULD generate a MIME entity with "multipart/alternative" for the value of its Content-Type header and three child entities, as follows:<74>

1. The first child entity has "text/plain" for the value of its Content-Type header. Its content SHOULD be plain text generated from the value of the PidTagRtfCompressed property, but can instead be copied from the value of the PidTagBody property, assuming that the two are equal.

2. The second child entity has "text/HTML" for the value of its Content-Type header. Its content SHOULD be HTML text generated from the value of the PidTagRtfCompressed property, but MAY instead be copied from the value of the PidTagHtml property, assuming that the two are equal.<75>

3. The third child entity has "text/calendar" for the value of its Content-Type header, and its content MUST be generated as specified in [MS-OXCICAL].<76>

#### 2.1.3.8.3 Calendar Message with Inline Attachments

When the best body format type of a calendar item or meeting message is RTF and there are inline attachments, MIME writers SHOULD generate a MIME entity with "multipart/related" for the value of its Content-Type header and two or more child entities, as follows:<77>

1. The first child entity is a multipart/alternative structure generated as specified in section 2.1.3.8.2.

2. Subsequent child entities are generated from the message object's inline attachments. Each entity MUST be generated as specified in section 2.1.4.1.

### 2.1.4 Attachments

Each Attachment object in a message object represents one attachment. MIME writers SHOULD classify Attachment objects based on the value of PidTagAttachMethod property, as specified in the following table.

| PidTagAttachMethod value | Attachment object Classification |
|---|---|
| 5 | Embedded message attachments |
| 6 | OLE attachments |
| All other values | Ordinary file attachments |

Note that ordinary file attachments can contain additional Macintosh-specific data. These attachments require special handling, as specified in section 2.1.4.3.

MIME writers SHOULD generate a **vCard** 3.0 attachment when generating **contact** information in a MIME message, as specified in section 2.1.4.6.

Additionally, MIME writers SHOULD classify Attachment objects as inline or not inline, as specified in section 2.1.4.1.

MIME writers SHOULD generate different MIME structures for the message depending on the presence of inline and non-inline attachments, as specified in the following three examples:

If both inline and non-inline attachments are present, MIME writers SHOULD generate the structure shown in figure 1. For an example of this structure, see section 3.1.1.

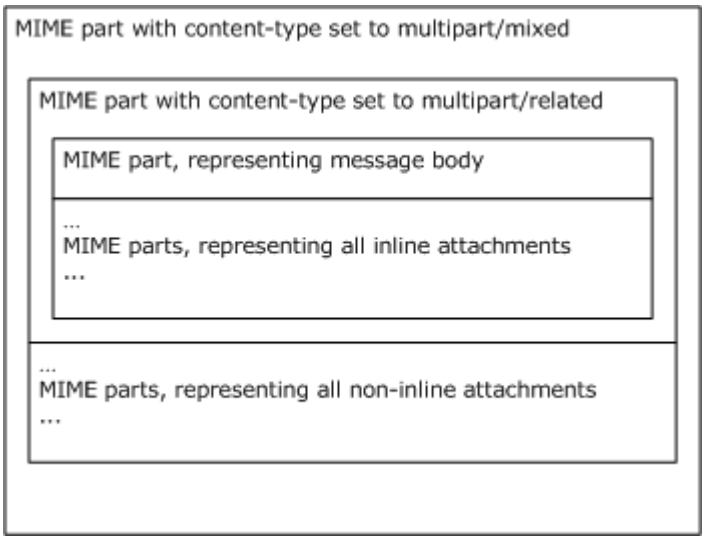


**Figure 1: Inline and non-inline attachments present**

If only inline attachments are present, MIME writers SHOULD generate the structure shown in figure 2. For an example of this structure, see section 3.1.2.

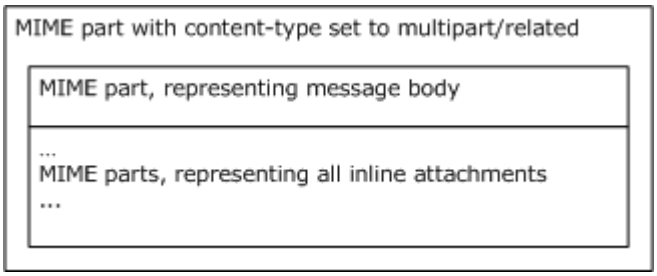


**Figure 2: Only inline attachments present**

If only non-inline attachments are present, MIME writers SHOULD generate the structure shown in figure 3. For an example of this structure, see section 3.1.3.

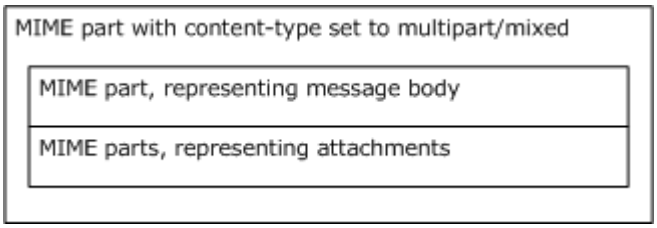


**Figure 3: Only non-inline attachments present**

#### 2.1.4.1 Inline Attachments

Clients SHOULD NOT create inline attachments if the best body text format is plain text. MIME writers SHOULD ignore indications that an attachment is inline for plain text messages by ignoring the presence of these properties on the attachment:

- PidTagAttachFlags
- PidTagAttachContentId
- PidTagAttachContentLocation

Likewise, clients SHOULD NOT designate attached Message objects as inline, and MIME writers SHOULD NOT treat attached Message objects as inline.<78>

##### 2.1.4.1.1 Inline Attachments in RTF Messages

If the best body text format is RTF, MIME writers SHOULD treat all OLE attachments, and only OLE attachments, as inline attachments.<79> OLE attachments have 0x00000006 for the value of PidTagAttachMethod.

RTF text does not contain explicit references to inline attachments, as HTML text does. Instead, the position of an inline attachment in the RTF text is indicated by an "\objattph" tag; clients insert such a tag into the RTF text for each inline attachment, as specified in [MS-OXRTFEX]. Clients also set the value of the PidTagRenderingPosition property to indicate the order of inline attachments: the attachment with the lowest value of this property matches the first "\objattph" tag; the next lowest matches the second "\objattph" tag, and so on. MIME writers sort inline attachments by the value of PidTagRenderingPosition when converting RTF text with inline attachments to HTML, and insert an HTML IMG tag into the generated HTML at the position corresponding to the RTF "\objattph" tag.

##### 2.1.4.1.2 Inline Attachments in HTML Messages

To mark an Attachment object in a message the best body text format for which is HTML as inline, clients do the following:

1. Set bit 3 (0x00000004) in the value of the Attachment object's PidTagAttachFlags property to TRUE.
2. Set the value of either PidTagAttachContentId (preferred) or PidTagAttachContentLocation on the Attachment object. If PidTagAttachContentLocation is used, PidTagAttachContentBase MAY be set to fully qualify a relative URI in PidTagAttachContentLocation. For details, see [RFC2557].
3. Include a tag that refers to the URI specified in (2) in the HTML message text. If PidTagAttachContentId is used, the URI MUST use the "cid:" scheme.<80>

MIME writers SHOULD NOT rely entirely on bit 0x00000004 of the PidTagAttachFlags property value to be set correctly for all attachments. Instead, MIME writers SHOULD verify all three conditions specified when deciding whether to treat an attachment as inline. <81><82>

#### 2.1.4.2 Attached Files

This section concerns generating attachments for **pure MIME messages**. When generating a TNEF message, all attachment data is written to the TNEF body part, as specified in [MS-OXTNEF].

#### 2.1.4.2.1 File Name

For the file name in a MIME representation of an attached file, MIME writers SHOULD use the value of the PidTagAttachLongFilename property. If this value is not available, MIME writers SHOULD use the value of the PidTagAttachFilename property, and can use an empty string if this value is also not available. The attached file name SHOULD be written to several different MIME headers, as specified in the next section.

If a file extension is needed for mapping the attachment content type, it SHOULD be obtained by copying all characters after the last "." (U+002E) character in the file name.

#### 2.1.4.2.2 Content-Type, Content-Description, Content-Disposition

MIME writers SHOULD determine the primary value of the Content-Type header for an attached file by using the following steps:

1. Acquire the value of the PidTagAttachMimeTag property.<83> If this value is not available, MIME writers determine the Content-Type by mapping it from the file extension (which is determined from the attachment file name, as specified in section 2.1.4.2.1), or by examining the file content itself. As a last resort, the MIME writer uses "application/octet-stream".

2. If the value acquired in the previous step does not match requirements for **MIME content-type**, as specified in [RFC2045], or if the value represents any multipart Content-Type, or if the value matches one of the following values, MIME writers replace it with "application/octet-stream":

   - application/applefile
   - application/mac-binhex40
   - message/rfc822

The value acquired as a result is then used as the value of the Content-Type MIME header. MIME writers SHOULD also generate the name parameter for this header, by using the attachment file name (determined as specified in section 2.1.4.2.1) as a value.

MIME writers SHOULD generate a Content-Description header by using the value of the PidTagDisplayName property.<84> If the property has no value, an empty header can be generated. If any of the conditions specified in [RFC2047] section 1 apply, the value of the Content-Description header SHOULD be encoded as specified in that document.<85>

The value for Content-Disposition header SHOULD be generated based on whether the attachment is inline or not, as specified in section 2.1.4.1. For inline attachments, the value is "inline", and for non-inline attachments, the value is "attachment".<86>MIME writers SHOULD generate the following parameters for this header:

- filename: the attachment file name determined as specified in 2.1.4.2.1 is used as a value.
- size: PidTagAttachSize property value SHOULD be used as a parameter value. The size parameter SHOULD be generated only if this property value is available and greater than 0 (zero).<87><88>
- creation-date: PidTagCreationTime property value SHOULD be used as the parameter value; if the property value is not available, the current time SHOULD be used. In either case, the creation time SHOULD be converted from UTC to a local time zone of the MIME writer's choice and formatted as specified in [RFC2822].<89><90><91>

- modification-date: PidTagLastModificationTime property value SHOULD be used as the parameter value; if the property value is not available, the current time SHOULD be used. In either case, the modification time SHOULD be converted from UTC to a local time zone of the MIME writer's choice and formatted as specified in [RFC2822].<92><93><94>

#### 2.1.4.2.3 Content-ID, Content-Location, Content-Base

MIME writers SHOULD generate a Content-ID MIME header if the value of the PidTagAttachContentId contains non-whitespace characters. All trailing and leading whitespace characters SHOULD be removed from this value. If the resulting value does not start with "<" (U+003C), or does not end with ">" (U+003E), it SHOULD be enclosed in angle brackets. The resulting string becomes the value of the Content-ID header.

MIME writers SHOULD generate a Content-Location MIME header if the PidTagAttachContentLocation property contains a value that is a valid URI. This value SHOULD be copied to the value of the Content-Location header.

MIME writers SHOULD generate a Content-Base MIME header if the PidTagAttachContentBase property contains a value that is a valid absolute URI. This value SHOULD be copied to the value of the Content-Base header.

#### 2.1.4.2.4 Content-Transfer-Encoding, MIME Part Body

MIME writers SHOULD use base64 (see [RFC2045]) as encoding for all ordinary file attachment MIME part bodies. As specified in [RFC2045], this also means that MIME writers SHOULD correspondingly generate the Content-Transfer-Encoding MIME header, and set its value to "base64".

MIME writers use the value of the PidTagAttachDataBinary property to generate the MIME entity body for this attachment. If the property does not exist or has 0 (zero) length, an empty MIME entity body SHOULD be generated.

### 2.1.4.3 MacBinary Attached Files

For interoperability with Macintosh-based mail clients, message attachments in MIME can be encoded by using one of the following Content-Types:

- application/applefile, as specified in [RFC1740].
- application/mac-binhex40, as specified in [RFC1741].
- multipart/appledouble, as specified in [RFC1740].

MIME writers SHOULD generate multipart/appledouble, as this MIME type is recommended by [RFC1740] for use in most cases.<95>

As specified in [RFC1740], the multipart/appledouble **MIME part** contains two sub-parts: a header part, with a Content-Type of "application/applefile", and a data part that contains actual file data (with Content-Type set to the value that corresponds to the actual MIME type of the file that is encoded).

To trigger encoding of an Attachment object as multipart/appledouble, clients set property values on the Attachment object as follows:

1. The value of the PidTagAttachMethod property is "0x00000001" (file attachment).

2. The value of the PidTagAttachEncoding property is the following byte string (expressed in hexadecimal): "%x2A.86.48.86.F7.14.03.0B.01".

3. The attachment content, which is the value of the PidTagAttachDataBinary property, is encoded in MacBinary format.

MacBinary is a way of serializing all attributes of a Macintosh file, including both data and resource forks, into a single stream. The MacBinary format elements relied upon in this specification are summarized very briefly by the following two tables. What follows is intended to specify server behavior with respect to MacBinary data; it is not normative with respect to the MacBinary format itself.

| MacBinary data field | Length | Description |
|---|---|---|
| MacBinary header | 128 bytes. | See more detail later in this section. |
| Secondary header data | Length is specified in bytes 120:121 of MacBinary header. | SHOULD be ignored by MIME writers.<96><97> |
| Data fork | Length is specified in bytes 83:86 of MacBinary header; begins on an even multiple of 128 bytes. | Contents of the file. |
| resource fork | Length is specified in bytes 87:90 of MacBinary header; begins on an even multiple of 128 bytes. | **Resources** associated with the file. |
| Get Info comment | Length is specified in byte 99 of MacBinary header. | SHOULD be ignored by MIME writers.<98><99> |

| Byte offset and length | Value |
|---|---|
| Byte 0 | Old version number, MUST be zero. |
| Byte 1 | Length of file name, MUST be less than 64. |
| Bytes 2 : 64 | File name, in us-**ASCII** charset; characters beyond the length specified in byte 1 MUST be ignored.<100> |
| Bytes 65 : 68 | File type, signed integer. |
| Bytes 69 : 72 | File creator, signed integer. |
| Byte 74 | Pad, MUST be 0 (zero). |
| Byte 82 | Pad, MUST be 0 (zero). |
| Bytes 83 : 86 | Data fork length, signed 32-bit integer in **big-endian** format. |
| Bytes 87 : 90 | Resource fork length, signed 32-bit integer in big-endian format. |

MIME writers MUST create a MIME entity with a Content-Type value of "multipart/appledouble", as specified in [RFC1740]. MIME writers SHOULD NOT write a name parameter for the Content-Type header in this MIME part. (This parameter is optional, as specified in [RFC1740].) As specified in [RFC1740], all additional information (other than the file contents) for a file that is to be transmitted

by using the multipart/appledouble MIME content-type SHOULD be put into a sub-part with Content-Type application/applefile.

If the Attachment object's PidNameAttachmentMacInfo property has a value, MIME writers MUST use it as the body of the application/applefile body part. The value of this property SHOULD be application/applefile data, as specified in [RFC1740] and further detailed in section 2.2.4.2.2, but containing only the header and resource fork sections.

If the Attachment object's PidNameAttachmentMacInfo property has no value, MIME writers SHOULD generate the body of the application/applefile body part from the resource fork and header data present in the MacBinary structure from the PidTagAttachDataBinary property, by using the mappings specified in section 2.2.4.2.2.

This MIME part is written out in the same way as in the case of an ordinary file attachment, with the following exceptions:

1. MIME writers MUST generate this part's MIME body by extracting only the file's data fork from the MacBinary structure in the PidTagAttachDataBinary property on the attachment, instead of just using raw data from this property.

2. MIME writers SHOULD copy the value of the PidNameAttachmentMacContentType property to the attachment body part's Content-Type header.

If PidNameAttachmentMacContentType has no value, MIME writers SHOULD write Content-Type: "application/octet-stream". An application/octet-stream type SHOULD also be written if PidNameAttachmentMacContentType has one of the following values:

- message/rfc822
- application/applefile
- application/mac-binhex40
- any multipart content-type

### 2.1.4.4 OLE Attachments

This section describes the generation of MIME entities that correspond to OLE attachments. An Attachment object is an OLE attachment if its PidTagAttachMethod property is set to 0x00000006.

MIME writers SHOULD generate a MIME part with "image/jpeg" for the value of its Content-Type MIME header to represent an OLE attachment in MIME. MIME writers SHOULD generate a description string for an OLE attachment, by using the value of the PidTagDisplayName property, but ensuring that this value ends with ".jpg".<101>The description string SHOULD be used as the name parameter of the Content-Type MIME header, and the value of the Content-Description MIME header SHOULD be generated with the same value.

A Content-Description header SHOULD be generated in the same way as for ordinary file attachments, with the following **exceptions**:<102>

1. The size parameter SHOULD NOT be generated.

2. The filename parameter value SHOULD be set to description string (see section 2.1.4.2.1).

The rest of MIME part headers SHOULD be generated in the same way as for ordinary file attachments, as specified in section 2.1.4.2.

OLE attachments SHOULD NOT have the PidTagAttachDataBinary property set, so MIME part body cannot be generated in the same way as for ordinary file attachments. Instead, the PidTagAttachDataObject property SHOULD be used. This property SHOULD contain a static rendition of an OLE object in **metafile** format, as specified in [MS-WMF]. MIME writers SHOULD use this data to generate a JPEG image that represents this OLE object, and generate the MIME part body by using this image data. If image generation fails, the server SHOULD use a generic icon representing an attachment.<103>

#### 2.1.4.5 Embedded Message Attachments

This section describes the generation of MIME entities that correspond to embedded message attachments. An attachment is considered by MIME writers to be an embedded message attachment if the value of its PidTagAttachMethod property is "0x00000005". MIME writers SHOULD generate a MIME entity with the Content-Type header set to "message/rfc822" (without parameters being generated). No other MIME headers SHOULD be generated. Instead, MIME writers SHOULD use properties of the embedded message to generate a pure MIME representation of this message, exactly as specified for ordinary messages, and use this data as the content of the message/rfc822 MIME entity. This MIME representation SHOULD be generated exactly as specified for ordinary messages, with the following exception: when writing MIME message headers by using PS_INTERNET_HEADERS properties, as specified in section 2.1.2.4, properties whose names begin with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-" SHOULD NOT be excluded from MIME generation (as they are for ordinary messages).

#### 2.1.4.6 vCard Generation

To generate a vCard attachment, clients attach a **contact object**, as specified in [MS-OXOCNTC], to a message. For MIME writers, an attached message with a PidTagMessageClass value that begins with "IPM.Contact" is the trigger for generating a vCard attachment in MIME. The vCard format is generated from contact object properties, as specified in [MS-OXVCARD]. vCard information is included in outbound MIME messages as a file attachment, as specified in section 2.1.4.2.2.

The vCard MIME part MUST use a Content-Type of "text/directory" with a profile of "vCard".<104>The charset is set to UTF-8. The vCard content uses quoted-printable encoding ([RFC2045] section 6.7).

### 2.1.5 Generating Pure MIME Messages

Pure MIME messages are generated by the MIME writer by combining stored message object content with the contents of the PidTagMimeSkeleton property, which is specified in section 2.4.1.<105> <106> The purpose of the MIME skeleton is to make the MIME output more accurately resemble the original MIME message when using the messaging server as the conduit between two MIME-based protocols, such as SMTP and POP or IMAP. The MIME skeleton contains the MIME structure and the headers of the original MIME message without any of the body part content, with some exceptions, as specified in section 2.4.

To generate a pure MIME message, the MIME writer reads the contents of PidTagMimeSkeleton associated with the message object, and then generates the pure MIME message by combining the contents of the skeleton with the saved message object content and attachments. The coupling of the generated MIME message to the original saved message by using the skeleton enables a more accurate reproduction of the MIME message provided by the server to MIME clients. For more details about how inbound MIME content is stored and saved, see section 2.4.

The behavior specified in this section applies only if **PidTagMimeSkeleton** is defined. If it is not, **MIME writers** MUST follow the guidelines set forth in sections 2.1.1, 2.1.2, 2.1.3, and 2.1.4, in that order.

#### 2.1.5.1 Generation Process

Generating a pure MIME message using the saved message object contents in conjunction with PidTagMimeSkeleton follows these steps.

1. The contents of PidTagMimeSkeleton are combined with the saved best body and message attachments from the Message object, using the order of the headers from PidTagMimeSkeleton to place the message contents in the original order in the generated message.

    1. Use the Content-ID (or X-ExchangeMime-Skeleton-Content-Id) header in PidTagMimeSkeleton to map the **MIME** attachment to the message body part with a matching value of **PidTagAttachContentId**.

    2. Use the Content-ID (or X-ExchangeMime-Skeleton-Content-Id) header to map the MIME body part to the message body part with a matching value of **PidTagAttachContentId**.

    3. The following headers are ignored in the skeleton and are regenerated from message object properties:<107>

        - Keywords
        - Importance
        - Priority
        - X-MsMail-Priority
        - X-Priority
        - X-Message-Flag

2. Encode attachment content using the Content-Transfer-Encoding value specified in the headers for the attachment in PidTagMimeSkeleton.

    - If the specified Content-Transfer-Encoding is not supported by the server, then use the default base64 encoding for the attachment content, and modify the value of Content-Transfer-Encoding in PidTagMimeSkeleton and in the outbound message to reflect this change.

3. The X-ExchangeMime-Skeleton-Content-Id headers SHOULD be deleted from the generated MIME message.

### 2.2 MIME Analysis

This section specifies both conversion from pure MIME to Message objects, and from TNEF to Message objects. The agent that performs the conversion is referred to as a MIME reader for clarity, because both clients and servers perform this conversion for different protocols.

As a general rule, when data occurs both in MIME and in a TNEF body part, the version found in MIME is to be preferred.<108> The message body is an exception to this **rule**: the plain text rendering found in MIME SHOULD NOT be used in preference to a richer (HTML or RTF) rendering found in TNEF. As an implementation guideline, MIME readers can process the TNEF body part before processing the remaining MIME data so that data from MIME overwrites the conflicting data from TNEF.

### 2.2.1 Address Elements

Most MIME address elements correspond to a group of four properties in the message object. The MIME address element itself has three parts, as specified in [RFC2822]: display name, comment, and e-mail address. The four properties are **DisplayName**, **EmailAddress**, **AddressType**, and **EntryID**. For a recipient in a message object, the four properties are referred to as the recipient property group, the members of which are the following:

- PidTagDisplayName
- PidTagEmailAddress
- PidTagAddressType
- PidTagEntryId

For other address elements in a message object, the four properties are grouped by name. For example, the four properties that correspond to the From header are the following:

- PidTagSentRepresentingName
- PidTagSentRepresentingEmailAddress
- PidTagSentRepresentingAddressType
- PidTagSentRepresentingEntryId

Collectively, these four properties are referred to as the PidTagSentRepresenting property group.

#### 2.2.1.1 Mapping Internet E-Mail Address Elements to a Property Group

In general, MIME readers map the three elements of an Internet e-mail address to the four properties as follows. The comment part of the Internet e-mail address SHOULD be ignored. Property names are written as "*DisplayName" to indicate that this algorithm applies to that member of any property group.

- *DisplayName: If the Internet e-mail address has a **display name** part, convert it to a Unicode string, performing decoding as specified in [RFC2047] if required, and write it to this property value. If there is no display name part, use the Internet e-mail address part.
- *AddressType: First check whether the Internet e-mail address was encoded by using IMCEA **encapsulation** (see section 2.1.1.8). <109>If it is, perform de-encapsulation (section 2.2.1.2) to obtain the Internet e-mail address and type, and write the type to this property. Otherwise, write "SMTP" to this property value. If there is no Internet e-mail address part, do not set this property value.
- *EmailAddress: If the Internet e-mail address was IMCEA-encapsulated, use the Internet e-mail address obtained by de-encapsulation. Otherwise, convert the entire Internet e-mail address part to Unicode and write it to this property value. If there is no Internet e-mail address part, do not set this property value.
- *EntryID: If there is an Internet e-mail address part, after the de-encapsulation step, perform a lookup against the address book for an entry any of whose proxy addresses matches this address. If an entry is found, construct an address book **entry ID** from that entry's DN, as specified in [MS-OXCDATA]. If no entry is found, construct a one-off EntryID from the display name, address type, and Internet e-mail address property values, according to the one-off EntryID specification in [MS-OXCDATA].

#### 2.2.1.2 Recognizing and De-Encapsulating IMCEA-Encapsulated Addresses

For details about IMCEA encapsulation, see section 2.1.1.8. De-encapsulation SHOULD be attempted only if the domain part of the encapsulated address is recognized as local, or otherwise able to deliver mail to the de-encapsulated address. <110>

An IMCEA-encapsulated SMTP address consists of the following six elements:

1. The literal string "IMCEA" in any combination of upper or lowercase letters.
2. The original address type, one or more ASCII characters.
3. A literal hyphen character, U+002D.
4. The encoded original address. Legal characters are upper and lower case ASCII letters, digits, hyphen (U+002D), equal sign (U+003D), underscore (U+005F), and plus sign (U+002B). Any other characters MUST be encoded as a plus sign (U+002B) followed by two hex digits.
5. A literal "@" sign, U+0040.
6. The encapsulation domain, such as "example.com".

To identify an e-mail address as IMCEA-encapsulated, it is sufficient to match items 1-3.

To obtain the original e-mail address and type from an encapsulated address, use the following procedure:

1. Copy item 2 to the e-mail address type.
2. Extract item 4, the encoded e-mail address.
3. Decode item 4 by replacing any underscore (U+005F) with a forward slash (U+002F), and replacing any sequence of plus sign (U+002B) followed by two hex digits with the single character the hex value for which is those two digits.

#### 2.2.1.3 From

To set the value of the **PidTagSentRepresenting** property group, MIME clients MUST set the From header value, as specified in [RFC2822].

MIME readers MUST set the value of the **PidTagSentRepresenting** property group to the value of the first e-mail address component of the From header (which can contain multiple e-mail addresses). If the From header contains multiple addresses, the first address MUST be used; the others are ignored.

When reading TNEF, MIME readers SHOULD use a From header value specified in MIME in preference to the **attSentFor** attribute or the **PidTagSentRepresenting** values of the attMsgProps attribute specified in TNEF, except for messages attached to a TNEF message, where a MIME header does not exist.<111>

#### 2.2.1.4 Sender

To set the value of the PidTagSender property group, MIME clients MUST set either the Sender or the From header value, as specified in [RFC2822].

MIME readers set the value of the PidTagSender property group to the value of the Sender header, if the Sender header is present in the MIME header. Otherwise, protocol servers SHOULD set the PidTagSender property group to the value of the first [RFC2822] mailbox of the From header.

When processing TNEF, MIME readers SHOULD use values specified in MIME in preference to the **attFrom** attribute or the PidTagSender property group values of the **attMsgProps** attribute specified in TNEF, except for messages attached to a TNEF message, where a MIME header does not exist.

#### 2.2.1.5 To, Cc, Bcc

To set the value of a recipient property group, MIME clients MUST set one of the To, Cc, or Bcc header values, as specified in [RFC2822], that corresponds to the recipient type, as specified in the following table.

| PidTagRecipientType value | Recipient type |
|---|---|
| 0x00000001 | To |
| 0x00000002 | Cc |
| 0x00000003 | Bcc |

MIME readers MUST add one recipient to the message object for each address in the To, Cc, and Bcc headers. MIME readers map the value of the recipient property group from address elements, as specified in section 2.2.1.1. Clients can specify multiple To, Cc, or Bcc headers, and MIME readers SHOULD process all of them.

MIME readers set the value of the PidTagRecipientType property for each recipient row to the value specified in the table.

When processing TNEF, MIME readers SHOULD use values specified in MIME in preference to the value of the attRecipTable attribute specified in TNEF, except for TNEF DSN messages and any messages attached to a TNEF message.

#### 2.2.1.6 Reply Recipients

To set the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties, MIME clients MUST set the Reply-To header value, as specified in [RFC2822].

Note that because Reply-to is an **address list** and not a single address, the property mapping is not a normal four-property group.

MIME readers set the values of the PidTagReplyRecipientEntries and the PidTagReplyRecipientNames properties (as specified in MS-OXOMSG) by mapping addresses from the Reply-To header.

When processing TNEF, MIME readers SHOULD use a Reply-To header value specified in MIME in preference to PidTagReplyRecipientEntries and the PidTagReplyRecipientNames values of the attMsgProps attribute specified in TNEF (except for messages attached to a TNEF message, where the MIME counterpart is not available).

#### 2.2.1.7 Disposition Notification Recipients

To set the value of the PidTagReadReceiptRequested and the PidTagReadReceipt property group, MIME clients MUST set the Disposition-Notification-To header value, as specified in [RFC3798].

MIME readers set the value of the PidTagReadReceiptRequested property to TRUE if the MIME header contains the Disposition-Notification-To header.

MIME readers map the value of the PidTagReadReceipt property group from the value of the Disposition-Notification-To header, if the field exists.

When processing TNEF, MIME readers SHOULD use a Disposition-Notification-To header value specified in MIME in preference to the PidTagReadReceiptRequested and **PidTagReadReceipt** property group values of the **attMsgProps** attribute specified in TNEF (except for messages attached to a TNEF message, where the MIME counterpart is not available).

#### 2.2.1.8 Return-Receipt-To

To set the value of the PidTagOriginatorDeliveryReportRequested property, MIME clients MUST set the [non-standard] Return-Receipt-To header value.

MIME readers set the value of the PidTagOriginatorDeliveryReportRequested property to TRUE if the message contains the Return-Receipt-To header. The actual value of the header is ignored, and receipts will be returned to the sender.

When processing TNEF, MIME readers SHOULD use a Return-Receipt-To header value specified in MIME in preference to the PidTagOriginatorDeliveryReportRequested property value of the **attMsgProps** attribute specified in TNEF (except for messages attached to a TNEF message, where the MIME counterpart is not available).

### 2.2.2 Envelope Elements

Many MIME headers that map directly to message object properties have string values. Unless otherwise specified, the string values are copied directly. All string values SHOULD be converted to Unicode (UTF-16) before they are copied to property values, and where applicable, the decoding specified in [RFC2047] is applied before generating the Unicode characters.

If there are multiple instances of a header, MIME readers SHOULD use the first instance to set the value of the corresponding property. <112> However, in the case of multiple recipient fields, MIME readers SHOULD combine the content of all instances to set the value of the corresponding property.

#### 2.2.2.1 MessageID

To set the value of the PidTagInternetMessageId property, MIME clients MUST set the Message-ID header value, as specified in [RFC2822]. MIME readers copy the value of the Message-ID header to the PidTagInternetMessageId property.

#### 2.2.2.2 Sent time

To set the value of the PidTagClientSubmitTime property, MIME clients MUST set the Date header value, as specified in [RFC2822].

MIME readers set the value of the PidTagClientSubmitTime property to the value of the Date header, converted to UTC. Full precision of the Date header, including seconds, MUST be preserved. If the Date header is missing or contains an invalid value, MIME readers set the value of the PidTagClientSubmitTime property to the current UTC time.

When processing TNEF, MIME readers use a Date header value specified in MIME in preference to an attDateSent or PidTagClientSubmitTime value specified in TNEF.

### 2.2.2.3 References

To set the value of the PidTagInternetReferences property, MIME clients write the value to a References header.

MIME readers copy the value of the References header to the value of the PidTagInternetReferences property. MIME readers MAY truncate the value of the PidTagInternetReferences property if it exceeds 64 KB in length.

### 2.2.2.4 Sensitivity

To set the value of the PidTagSensitivity property to a value other than normal, MIME clients MUST write the value to a Sensitivity header.

MIME readers map Sensitivity header values to PidTagSensitivity values as specified in the following table.

| PidTagSensitivity value | Sensitivity header value |
|---|---|
| 0x00000000 | Normal |
| 0x00000001 | Personal |
| 0x00000002 | Private |
| 0x00000003 | Company-Confidential |

### 2.2.2.5 Importance

To set the value of the PidTagImportance property, MIME clients SHOULD write the value to an Importance header.

MIME readers map Importance header values to PidTagImportance values as specified in the following table.

| Importance header value | PidTagImportance value |
|---|---|
| Low | 0x00000000 |
| Normal | 0x00000001 |
| High | 0x00000002 |

MIME clients MAY use a Priority, X-Priority, or X-MSMail-Priority header instead of an Importance header to set the value of the PidTagImportance property.<113> In that case, MIME readers map the header values to PidTagImportance values, as specified in the following tables. However, if an Importance header is present, MIME readers SHOULD use its value in preference to any of the others.

| Priority header value | PidTagImportance value |
|---|---|
| Non-Urgent | 0x00000000 |
| Normal | 0x00000001 |
| Urgent | 0x00000002<114> |

| X-Priority header value | PidTagImportance value |
|---|---|
| 5 | 0x00000000 |
| 4 | 0x00000000 |
| 3 | 0x00000001 |
| 2 | 0x00000002 |
| 1 | 0x00000002 |

| X-MSMail-Priority header value | PidTagImportance value |
|---|---|
| Low | 0x00000000 |
| Normal | 0x00000001 |
| High | 0x00000002 |

#### 2.2.2.6 Subject

To set the value of the PidTagSubjectPrefix and PidTagNormalizedSubject properties, MIME clients MUST set the Subject header value, as specified in [RFC2822].

MIME readers SHOULD analyze the Subject header value into a prefix and a normalized subject value, as specified in section 2.2.2.6.1, and then set the values of the PidTagSubjectPrefix and PidTagNormalizedSubject properties, rather than simply setting the value of PidTagSubject. MIME readers can truncate the Subject value.

MIME readers use a Subject header value specified in MIME in preference to an **attSubject** or PidTagSubject value specified in TNEF. They SHOULD, however, use PidTagSubjectPrefix and PidTagNormalizedSubject values from TNEF when they match the MIME subject, because of limitations in the subject normalization algorithm of section 2.2.2.6.1.

##### 2.2.2.6.1 Normalizing the Subject

If no values are available for PidTagNormalizedSubject and PidTagSubjectPrefix in the MIME message, protocol servers SHOULD parse the Subject property value and set those values as follows. If the Subject header value consists of one, two, or three characters (exclusive of colon (U+003A), blank (U+0020), or digits (U+0030 through U+0039)), followed by a colon (U+003A) and any number of blanks (U+0020), the protocol server SHOULD set the value of PidTagSubjectPrefix to the aforementioned one, two, or three characters appended with a colon and a space (": "), and SHOULD set the value of PidTagNormalizedSubject to the remainder of the Subject header value beginning immediately after the aforementioned blanks.

#### 2.2.2.7 Conversation Topic

To set the value of the PidTagConversationTopic property, MIME clients MUST write the value to a Thread-Topic header. This value SHOULD be the same as the value of the Subject header, normalized as specified in section 2.2.2.6.1 to remove any prefix.

MIME readers copy the value of a Thread-Topic header to the value of the PidTagConversationTopic property.<115> MIME readers SHOULD also use this header value as a hint to normalize the subject, as specified in section 2.2.2.6.1, if this value matches the tail of the Subject header value.

#### 2.2.2.8 Conversation Index

To set the value of the PidTagConversationIndex property, MIME clients MUST write the value to a Thread-Index header. The property data type is binary, and protocol clients encode the header value using base64 encoding, as specified in [RFC2045]. The format of the value is specified in [MS-OXOMSG].

MIME readers copy the value of a Thread-Index header to the value of the PidTagConversationIndex property, assuming the base64-encoded text can be successfully decoded to binary data. MIME readers SHOULD ignore a Thread-Index header that does not contain base64-encoded binary data.

#### 2.2.2.9 In-Reply-To Message ID

To set the value of the PidTagInReplyToId property, MIME clients MUST write the value to an In-Reply-To header, as specified in [RFC2822].

MIME readers copy the value of an In-Reply-To header to the value of the PidTagInReplyToId property.

#### 2.2.2.10 ReplyBy Time

To set the value of the PidTagReplyTime property, MIME clients MUST set the Reply-By header value, as specified in [RFC2156].

MIME readers MUST set the value of the PidTagReplyTime property to the value of the Reply-By header, converted to UTC time.

When processing TNEF, MIME readers MUST use a Reply-By header value specified in MIME in preference to a PidTagReplyTime value specified in TNEF.

#### 2.2.2.11 Language Properties

To set the value of the PidTagMessageLocaleId property, MIME clients MUST set the Content-Language header, as specified in [RFC3282].

MIME readers set the value of the PidTagMessageLocaleId property by extracting the first language tag from the value of the Content-Language header and mapping it to an LCID, as specified in [MS-LCID]. MIME readers SHOULD use the value of a Content-Language header in preference to the value of PidTagMessageLocaleId found in the attMsgProps attribute of a TNEF message.

To set the value of the PidNameAcceptLanguage property, MIME clients SHOULD write the value to an Accept-Language header. MIME clients MAY write an X-Accept-Language header instead.

MIME readers SHOULD copy the value of either header to the value of the PidNameAcceptLanguage property. If both headers are present, MIME readers SHOULD use the Accept-Language header.<116><117>

#### 2.2.2.12 Categories

To set the value of the PidNameKeywords property, MIME clients MUST set the Keywords header, as specified in [RFC2076].

MIME readers SHOULD map the value of a Keywords header to the value of the PidNameKeywords property by splitting the Keywords header value at each comma (U+0032), trimming whitespace, and storing each keyword as an individual value of the multiple string **property**.

To prevent conflicts among category schemes in different organizations, MIME readers MAY omit mapping the Keywords header to the PidNameKeywords property.

#### 2.2.2.13 Message Expiry Time

To set the value of the PidTagExpiryTime property, MIME clients MUST write the value to the Expires header.

MIME readers copy the value of the Expires header to the value of the PidTagExpiryTime property, after converting it to UTC time.

MIME clients MAY use an Expiry-Date header instead of an Expires header. Protocol servers MUST use the value of the Expires header in preference to Expiry-Date, if both headers are present.<118>

#### 2.2.2.14 Suppression of Automatic Replies

To set the value of the PidTagAutoResponseSuppress property to -1, indicating that all automatic replies to the message are to be suppressed, MIME clients SHOULD write an X-AUTO-Response-Suppress header with the value "All". MIME clients MAY, instead, write a Precedence header with any value.

To set the value of the PidTagAutoResponseSuppress property to a more specific value, MIME clients write an X-AUTO-Response-Suppress header with one or more values from the table in section 2.1.2.20 selected.

MIME readers SHOULD map individual elements of an X-Auto-Response-Suppress header to bits in the value of the PidTagAutoResponseSuppress property according to the table.<119> If both X-Auto-ResponseSuppress and Precedence headers are present, the PidTagAutoResponseSuppress property value SHOULD be 0xFFFFFFFF.<120> If the value of the X-Auto-Response-Suppress header is other than as specified in the table in section 2.1.2.20, MIME readers SHOULD ignore the entire header.<121><122>

#### 2.2.2.15 Content Class

To set the value of the PidNameContentClass property, MIME clients MUST write the value to a Content-Class header.<123>

MIME readers copy the value of a Content-Class header to the value of the PidNameContentClass property.

MIME readers SHOULD also set the value of the PidTagMessageClass property for certain Content-Class header values as specified in the following table, but only if the value of PidTagMessageClass would otherwise be set to "IPM.Note".<124>

| Content-Class header value | PidTagMessageClass property value |
|---|---|
| "fax" | "IPM.Note.Microsoft.Fax" |
| "fax-ca" | "IPM.Note.Microsoft.Fax.CA" |
| "missedcall" | "IPM.Note.Microsoft.Missed.Voice" |

| Content-Class header value | PidTagMessageClass property value |
|---|---|
| "voice-uc" | "IPM.Note.Microsoft.Conversation.Voice" |
| "voice-ca" | "IPM.Note.Microsoft.Voicemail.UM.CA" |
| "voice" | "IPM.Note.Microsoft.Voicemail.UM" |
| Starts with "urn:content-class:custom." | "IPM.Note.Custom.", followed by the value of Content-class header, with "urn:content-class:custom." prefix removed.<125><126> |

Additionally, if the Content-Class header value begins with "InfoPath.", then MIME readers SHOULD extract a substring from the header value beginning immediately after the prefix and ending at the end of the header value. If this string contains a period character (U+002E), and the first occurrence of this character is not the last one in the string, this string SHOULD be further separated into two substrings. The delimiting period is not included into either one of the substrings.

The first substring SHOULD be additionally checked to match the string format of a **GUID** string (see [MS-DTYP]). If this check succeeds, the second substring SHOULD be saved as a value of the PidLidInfoPathFormName property. In addition, the first substring SHOULD be appended to "IPM.InfoPathForm." and written to the value of the PidTagMessageClass property.

If a message that is being processed by a MIME reader is clear signed or opaque signed, as specified in [MS-OXOSMIME], the appropriate suffix (".SMIME.MultipartSigned" or ".SMIME") SHOULD be appended to the value of PidTagMessageClass.

### 2.2.2.16 Message Flagging

To set the value of the PidLidFlagRequest property, MIME clients MUST write the value to an X-Message-Flag header.

MIME readers copy the value of an X-Message-Flag header to the value of the PidLidFlagRequest property. In addition, when an X-Message-Flag header is present, MIME readers SHOULD do all the following:<127>

1. Set the value of the PidTagFlagStatus property to 2 (denoting that the message is flagged).
2. Copy the value of the PidTagSubject property to the value of the PidLidToDoTitle property.
3. Set the value of the PidLidTaskStatus property to 0 (zero) (denoting that a task is not started).
4. Delete or disregard any existing property values for the following properties:

   PidLidTaskDueDate

   PidLidTaskStartDate

   PidTagFlagCompleteTime

   PidLidTaskDateCompleted
5. Set the value of the PidLidTaskComplete property to FALSE.
6. Set the value of the PidLidPercentComplete property to 0.0.
7. Set the value of the PidTagToDoItemFlags property to 8.

### 2.2.2.17 List Server Properties

To set the values of list server–related properties, MIME clients MUST write headers as specified in the following table.

| Property | Preferred header name | Alternate header name |
|---|---|---|
| PidTagListHelp | List-Help | X-List-Help |
| PidTagListSubscribe | List-Subscribe | X-List-Subscribe |
| PidTagListUnsubscribe | List-Unsubscribe | X-List-Unsubscribe |

MIME readers copy header values to property values as specified in the table.<128>

### 2.2.2.18 Payload Properties

To set the value of the PidTagAttachPayloadClass or PidTagAttachPayloadProviderGuidString properties, MIME clients SHOULD write an X-Payload-Class and an X-Payload-Provider-GUID header, respectively. Such headers SHOULD be written to a MIME entity that will be analyzed as an attachment, as specified in section 2.2.4.<129>

MIME readers MUST copy these header values to the values of the corresponding properties.<130> <131> MIME readers SHOULD copy these headers when they appear on a MIME entity that is analyzed as a message or message body, rather than as an attachment.<132>

### 2.2.2.19 Purported Sender Domain

MIME readers copy the value of the X-MS-Exchange-Organization-PRD header to the PidTagPurportedSenderDomain **property** ([MS-OXPROPS] section 2.982).<133>

### 2.2.2.20 Sender Id Status

MIME readers copy the value of the X-MS-Exchange-Organization-SenderIdResult header to the PidTagSenderIdStatus **property** ([MS-OXPROPS] section 2.1116).<134> The values of the header are mapped to PidTagSenderIdStatus as follows:

| Symbolic Name | Value |
|---|---|
| Neutral | 0x00000001 |
| Pass | 0x00000002 |
| Fail | 0x00000003 |
| SoftFail | 0x00000004 |
| None | 0x00000005 |
| TempError | 0x80000006 |
| PermError | 0x80000007 |

### 2.2.2.21 Spam Confidence Level

MIME readers parse the value of the X-MS-Exchange-Organization-SCL header as an integer value in the range -1 to 10, and assigns that value to the PidTagContentFilterSpamConfidenceLevel **property** ([MS-OXPROPS] section 2.720).<135>

### 2.2.2.22 Classification Properties

In order to preserve full client/server data fidelity in the MIME content, if the PidLidClassified property is present in the mail object and is set to TRUE, then MIME clients SHOULD write the following header:<136>

```
X-Microsoft-Classified: true
```

In addition, MIME clients SHOULD write header values for all of X-Microsoft-Classification, X-Microsoft-ClassDesc, X-Microsoft-Classification-GUID, and X-Microsoft-Classification-Keep.

When the appropriate X-Microsoft-Classified header is present, MIME readers SHOULD map or copy all classification header values to their corresponding property values, as specified in the following table. If the X-Microsoft-Classified header is missing or has a value other than "true", MIME readers SHOULD NOT set any of the five property values listed in the table.<137><138>

| Classification header | Classification property | Header value mapping |
|---|---|---|
| X-Microsoft-Classified | PidLidClassified | "true" maps to TRUE. |
| X-Microsoft-ClassKeep | PidLidClassificationKeep | "true" maps to TRUE.<br>"false" maps to FALSE. |
| X-Microsoft-Classification | PidLidClassification | No mapping. The string value is copied directly. |
| X-Microsoft-ClassDesc | PidLidClassificationDescription | No mapping. The string value is copied directly. |
| X-Microsoft-ClassID | PidLidClassificationGuid | No mapping. The string value is copied directly. |

### 2.2.2.23 Unified Messaging Properties

To set the values of **unified messaging** properties, MIME clients SHOULD write the value to the corresponding header, as specified in the following table.<139>

| Header name | Property |
|---|---|
| X-CallingTelephoneNumber | PidTagSenderTelephoneNumber |
| X-VoiceMessageSenderName | PidTagVoiceMessageSenderName |
| X-AttachmentOrder | PidTagVoiceMessageAttachmentOrder |
| X-CallID | PidTagCallId |
| X-VoiceMessageDuration | PidTagVoiceMessageDuration; header value MUST be parsed as |

| Header name | Property |
|---|---|
| | PtypInteger32 |
| X-FaxNumberOfPages | PidTagFaxNumberOfPages; header value MUST be parsed as PtypInteger32 |

MIME readers SHOULD copy header values to property values, as specified in the table.<140> <141>

#### 2.2.2.24 Content-ID

To set the value of the PidTagBodyContentId property, MIME clients MUST write the value to a Content-ID header on a MIME entity that maps to a message body, as specified in section 2.2.3.

MIME readers SHOULD copy the value of a Content-ID header on such a MIME entity to the value of the PidTagBodyContentId property.<142><143>

MIME clients can write either a Content-ID or a Content-Location header, but SHOULD NOT write both on a single MIME entity.

#### 2.2.2.25 Content-Base

To set the value of the PidNameContentBase property, MIME clients MUST write the value to a Content-Base header on a MIME entity that maps to a message body, as specified in section 2.2.3.

MIME readers copy the value of a Content-Base header on such a MIME entity to the value of the PidNameContentBase property.

To set the value of the PidNameContentBase property, MIME clients SHOULD write the value to a Content-Base header on a MIME entity that maps to a Message object (top-level or attached).

MIME readers SHOULD copy the value of a Content-Base header on such a MIME entity to the value of the PidNameContentBase property.<144>

#### 2.2.2.26 Content-Location

To set the value of the PidTagBodyContentLocation property,<145><146> MIME clients SHOULD write the value to a Content-Location header on a MIME entity that maps to a message body, as specified in section 2.2.3.

MIME readers SHOULD copy the value of a Content-Location header on such a MIME entity to the value of the PidTagBodyContentLocation property.<147>

#### 2.2.2.27 XRef

MIME readers copy the value of an XRef header to the value of the PidNameCrossReference property.<148>

#### 2.2.2.28 PidTagTransportMessageHeaders

MIME readers SHOULD copy all headers, with certain exceptions, from an inbound message to the value of the PidTagTransportMessageHeaders property. With the exception of headers specifically mentioned in section 2.1.2, headers that begin with the reserved name prefixes "X-MS-Exchange-Organization-" and "X-MS-Exchange-Forest-" SHOULD NOT be copied to PidTagTransportMessageHeaders.

This property value SHOULD be set only by MIME readers upon delivery of a message from SMTP, in which case it SHOULD be set to the header of the **top-level message** (with exceptions as already specified).<149>

### 2.2.2.29 Generic Headers in PS_INTERNET_HEADERS

To create a **named property** in the PS_INTERNET_HEADERS property set, whose name is a header name and whose value is a header value, MIME clients MUST write the name and value to a header.

For each such header, MIME readers SHOULD create a named property as follows:<150><151><152>

- The PropertyName GUID is "%X86.03.02.00.00.00.00.00.C0.00.00.00.00.00.46".
- The PropertyName name is the header name.
- The property value is the header value. If the header value was encoded according to [RFC2047], MIME readers MUST decode it.

MIME readers MUST NOT create such named properties for any MIME header that is mapped to a different property, as specified elsewhere in this section. MIME readers SHOULD NOT create such named properties for any of the following MIME headers:

- Received
- Resent-From
- Resent-Sender
- Resent-Date
- Resent-Message-Id
- Content-Type
- Content-Disposition
- Content-Description
- Content-Transfer-Encoding
- Content-ID
- Content-MD5
- MIME-Version
- Return-Path
- Comments
- AdHoc
- Apparently-To
- Approved
- Control

- Distribution
- Encoding
- FollowUp-To
- Lines
- Bytes
- Article
- Supercedes
- NewsGroups
- NntpPostingHost
- Organization
- Path
- RR
- Summary
- Trace
- Encrypted
- X-MimeOle
- X-MS-TNEF-Correlator
- Any header the name of which begins with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-", except for the following:
  - X-MS-Exchange-Organization-AuthAs
  - X-MS-Exchange-Organization-AuthDomain
  - X-MS-Exchange-Organization-AuthMechanism
  - X-MS-Exchange-Organization-AuthSource

### 2.2.3 Body Text

Unlike MIME, which allows an arbitrary number of inline text body parts, message objects distinguish one text body part as the message body.

#### 2.2.3.1 Client Actions

To send the value of PidTagBody as the definitive body text, MIME clients SHOULD create a MIME message in which the first or only element has "text/plain" as the value of the Content-Type header, and that element's body contains the text. MIME clients SHOULD specify the charset of the message body text on the corresponding MIME element.

To send the value of PidTagHtml as the definitive body text, MIME clients SHOULD create a MIME message in which the first or only MIME element has "text/HTML" as the value of the Content-Type header, and that element's body contains a well-formed HTML document. Clients SHOULD generate a multipart/alternative structure with a text/plain representation, so that a greater number of clients can process the message.

To send the value of PidTagRtfCompressed as the definitive body text, MIME clients SHOULD create a MIME message that contains a TNEF body part, as specified at the beginning of section 2, and write the value of PidTagRtfCompressed into the attMsgProps attribute of the TNEF.

#### 2.2.3.2 Determining Which MIME Element Is the Message Body

The rules a MIME reader follows for selecting a message body are both qualifying, or positive, and disqualifying, or negative. To qualify as a message body, a MIME entity MUST meet at least one of the following **conditions**:

- Content-Type header value is "text/plain", "text/HTML", "text/enriched", or "text/calendar".
- Content-Type header value is "multipart/alternative" and at least one child MIME entity is "text/plain", "text/HTML", "text/enriched", or "text/calendar".
- Content-Type header value is "multipart/related", and its first child MIME entity is either "text/HTML", or "multipart/alternative" with at least one text/HTML child MIME entity.

To qualify as a message body, a MIME entity MUST NOT have a Content-Disposition header with the value "attachment".

In all cases, it is the text body part and not the containing multipart itself that is mapped to the message body.

MIME readers select the first MIME entity that qualifies according to the rules as the message body. MIME readers SHOULD then map the content of the selected MIME entity to a message object property value according to the following rules:

1. If the body MIME entity is a single text/plain, copy its content to the value of the PidTagBody property.
2. If the body MIME entity is a single text/HTML, copy its content to the value of the PidTagHtml property.
3. If the body MIME entity is a single text/enriched, convert its content to HTML and copy the result to the value of the PidTagHtml property.
4. If the body MIME entity is a single text/calendar, parse the iCalendar document and copy the value of the DESCRIPTION property to PidTagBody. If the DESCRIPTION property is missing, MIME readers can use the value of the COMMENT property instead. For details, see [MS-OXCICAL].
5. If the body MIME entity is multipart/alternative, MIME readers SHOULD select the last child entity that has one of the four eligible types ("text/plain", "text/HTML", "text/enriched", or "text/calendar") and map it. However, if the last child entity is text/calendar and one of the preceding entities is text/HTML, MIME readers SHOULD map the text/HTML, instead of the DESCRIPTION property of the text/calendar, to PidTagRtfCompressed.
6. If the body MIME entity is multipart/related, identify the first child MIME entity that is either text/HTML or multipart/alternative and map it according to rules 1-5.

#### 2.2.3.2.1 Selecting the Primary Message Text MIME Element

When alternative text MIME elements are present and eligible for use as the message body, as specified in section 2.2.3.2, MIME readers SHOULD choose a MIME element to populate the message body text by using the following content types, in descending order of preference:

- text/HTML
- text/enriched
- text/plain
- text/calendar (but only if the METHOD property value of the text/calendar body part is PUBLISH, REQUEST, REPLY, or CANCEL)

If text/HTML is selected, MIME readers copy the MIME element body text to the value of the PidTagHtml property, map the charset parameter of the MIME element's Content-Type header to a code page, and set the value of the PidTagInternetCodepage property to that code page. If the charset parameter is not present, MIME readers MAY use the value of a Content-Type meta tag in the HTML document, but SHOULD verify its validity before using it.

If text/plain is selected, MIME readers convert the plain text to UTF-16LE and write the resulting text to the value of the PidTagBody property. MIME readers SHOULD, in addition, map the value of the charset parameter of the MIME element's Content-Type header to a code page, and set the value of the PidTagInternetCodepage property to that code page.

If text/enriched is selected, MIME readers convert the MIME element body text to either text/plain, text/HTML, or RTF, and handle that as previously specified.

If both text/HTML and text/calendar body parts are present and eligible for use as message body, instead of writing text to the PidTagHtml property, MIME readers SHOULD convert the HTML text to RTF and write it to the value of the PidTagRtfCompressed property. Alternatively, MIME readers can choose to use plain text from a text/plain body part or from data in the text/calendar body part, as specified in [MS-OXCICAL]. MIME readers MUST NOT set the PidTagHtml property on a calendar or meeting message object.

### 2.2.4 Attachments

During MIME analysis, MIME readers classify all non-multipart MIME entities and multipart/appledouble MIME entities (that contain appropriate child MIME sub-parts) into the following three categories:

1. MIME entities that can potentially represent the message body, as specified in section 2.2.3.2.
2. MIME entities that represent non-inline attachments.
3. MIME entities that represent attachments that can potentially be inline.

All MIME entities that can be classified as attachments (2nd or 3rd category) SHOULD be treated by MIME readers as attachment MIME part, and an entry in an attachments table SHOULD be created for each such MIME part. However, depending on the value of the Content-Type MIME header, analysis SHOULD be done differently, as follows:

1. message/rfc822 MIME entities SHOULD be treated as embedded message attachments, as specified in section 2.2.4.3.

2. Multipart/appledouble, application/applefile, and application/mac-binhex40 MIME entities SHOULD be treated as Macintosh attachments, as specified in section 2.2.4.2.

3. Message/external-body attachments SHOULD be treated as external body attachments, as specified in section 2.2.5.

4. All other attachments SHOULD be treated as regular file attachments, as specified in section 2.2.4.1.

If no Content-Type header is present on a MIME entity, MIME readers SHOULD treat it as text/plain (unless this MIME entity is a sub-part of multipart/digest, in which case the default value for the Content-Type MIME header is message/rfc822).

### 2.2.4.1 Regular File Attachment MIME Part Analysis

When creating an Attachment object for a regular file attachment, MIME readers set the value of the PidTagAttachMethod property to 0x00000001.

#### 2.2.4.1.1 File name

The attachment file name SHOULD be determined by MIME readers in the following order:

1. If the Content-Disposition header exists on the attachment MIME entity, and a non-empty filename parameter is available on this header, the filename parameter value SHOULD be used.

2. Otherwise, if the Content-Type header is available on the attachment MIME entity, and a non-empty name parameter is available on this header, the name parameter value SHOULD be used.

3. Otherwise, if the Content-Transfer-Encoding header is set to "binhex", MIME readers SHOULD try to parse MIME part body as MacBinary structure, as specified in section 2.2.4.2.3. Only the first 128 bytes of the MIME body (decoded with binhex, see [RFC1741]) SHOULD be parsed. If parsing of MacBinary structure succeeds, file name data from this structure SHOULD be used.

4. Otherwise, if the attachment MIME part body is encoded with uuencode (see section 2.3.1 and [IEEE1003.1]), and it contains file name data, this file name SHOULD be used.

5. Otherwise, if the Content-Description header is available on the attachment and its value is non-empty, it SHOULD be used as the file name value for an attachment. (Even if a file name for an attachment was found in one of the previous steps, this value SHOULD be written to PidTagDisplayName for an attachment.)

MIME readers SHOULD sanitize the resulting file name and display name by removing characters that are not legal. Invalid characters are listed in the following table.<153>

| Description | Code point | Character |
|---|---|---|
| Control characters | U+0000 through U+001F<154> | |
| Double quote | U+0022 | " |
| Forward slash | U+002F | / |
| Colon | U+003A | : |
| Left angle bracket | U+003C | < |
| Right angle bracket | U+003E | > |

| Description | Code point | Character |
|---|---|---|
| Pipe | U+007C | \| |
| Backslash | U+005C | \ |

The following steps SHOULD then be applied both to the attachment file name and the display name (if the display name is not available, the empty string SHOULD be used):<155>

- Replace all Unicode separator characters with spaces. Unicode separator characters are specified in [UNICODE] section 6.2.
- Separate name into base and extension parts. The extension is defined as the trailing part of a name that starts after the last appearance of a "." character (U+002E) in the name, or an empty string if name contains no such character.
- Remove all leading and trailing spaces and leading and trailing "." (U+002E) characters from both the base and the extension.<156>

If the extension part of the display name is not empty and does not match the extension part of file name, it SHOULD be appended to the base part of display name.

If the file name base and/or file name extension is empty, the MIME reader SHOULD generate an attachment file name base and/or extension. The filename can be created using any file name generation convention that conforms to the filename guidelines specified in this section.<157>

After that, if the base part of the display name is empty, it SHOULD be replaced with the base part of the file name.<158> Finally, the file name base, file name extension, and display name SHOULD be reassembled from the base and extension parts and saved in the appropriate properties, as specified in the following table.

| Property | Value |
|---|---|
| PidTagDisplayName | <display name base>.<file name extension> |
| PidTagAttachLongFilename | <file name base>.<file name extension> |
| PidTagAttachExtension | .<file name extension> |

The value saved to PidTagAttachLongFilename SHOULD be further processed to form a valid **8.3 name**, and then written to PidTagAttachFilename, as follows:<159>

1. The value SHOULD be first separated into name and extension parts, using the last "." character (U+002E) as a separator. If no such character is present, or the only appearance of this character is in the beginning of the file name, the name part is considered to be empty but the extension is not empty; <160> the separator character itself is not included into the name or extension.
2. Replace the following characters with an underscore (U+005F): plus sign "+" (U+002B), comma "," (U+002C), equal sign "=" (U+003D), left square bracket "[" (U+005B), right square bracket "]" (U+005D), semicolon ";" (U+003B).<161><162>
3. Remove the following characters: space (U+0020), period "." (U+002E), apostrophe "'" (U+0027), asterisk "*" (U+002A), question mark "?" (U+003F), as well as characters with a UTF8 code greater than 127.<163>

4. If name is empty after removing such characters, MIME readers SHOULD generate a non-empty value.<164>

5. Trim the name part of the file name to 8 characters, and the extension part to 3 characters.<165>

6. If name was shortened, the name part SHOULD additionally be trimmed to 6 characters, and "~1" SHOULD be added to its end.<166><167>

7. Recombine the file name and extension, separated by a single "." (U+002E).

#### 2.2.4.1.2 Content Type

MIME readers SHOULD save the value of the Content-Type MIME header in the PidTagAttachMimeTag property during MIME analysis. The following notes apply for specific values of this header:

- The "application/ms-TNEF" value SHOULD be replaced with "application/octet-stream". This is in the rare case when a TNEF body part is corrupt and cannot be completely processed.<168>Ordinarily, a TNEF body part SHOULD NOT be written to an attachment, but analyzed into message object properties and discarded.
- The values "application/x-pkcs7-MIME" and "application/pkcs7-MIME": the entire Content-Type header value, including all parameter names and values, SHOULD be written to the PidNameContentType property value.
- For Content-Type values that start with "text/", if a charset parameter is present, the parameter value SHOULD be written to the PidTagTextAttachmentCharset property.

#### 2.2.4.1.3 Attachment Creation and Modification Date

If a Content-Disposition MIME header is present on the attachment MIME part, MIME readers SHOULD use its parameters to set creation and modification dates on the Attachment object.<169> If a parameter is missing or its value is not a valid date, the corresponding property value SHOULD NOT be set.<170> Date and time values MUST be translated to UTC.

| Content-Disposition parameter name | Property |
|---|---|
| creation-date | PidTagCreationTime |
| modification-date | PidTagLastModificationTime<171> |

#### 2.2.4.1.4 Attachment Content-Id, Content-Base, and Content-Location

If a Content-Id MIME header is present on the attachment MIME part, MIME readers SHOULD copy its value to the PidTagAttachContentId property. If this value starts with "<" (U+003C) and/or ends with '>' (U+003E), these characters SHOULD be removed.

If a Content-Location MIME header is present on the attachment MIME part, its value SHOULD be saved in the PidTagAttachContentLocation property.

If a Content-Base MIME header is present on the attachment MIME part, MIME readers SHOULD copy its value to the PidTagAttachContentBase property. <172>

Additionally, if an attachment MIME part is a child of a multipart/related MIME element, and either a Content-Id or Content-Location MIME header is present, MIME readers SHOULD mark the

attachment as inline, as specified in section 2.1.4.1.2. <173> MIME readers SHOULD verify whether the attachment is actually referenced from the message body, and mark it as inline only if that is the case; but MAY mark it as inline unconditionally. <174>

#### 2.2.4.1.5 Attachment Content-Transfer-Encoding and MIME Part Body

As specified in [RFC2045], a Content-Transfer-Encoding header might be present on the attachment MIME part. MIME readers SHOULD support the following values for this header:

- Base64. See [RFC2045]
- Quoted-printable. See [RFC2045]
- 7bit. See [RFC2045]
- 8bit. See [RFC3516]
- Binary. See [RFC3030]
- Mac-binhex40
- X-uuencode
- X-uue

As specified in [RFC2045], if the Content-Transfer-Encoding MIME header is missing, MIME readers MUST behave as if it were set to 7bit.

The attachment's content SHOULD be decoded by using the appropriate decoding procedure and saved as the value of the PidTagAttachDataBinary property. MIME readers SHOULD, as a rule, use RopOpenStream (as specified in [MS-OXCROPS]) to create this property value.

The encoding values mac-binhex40, x-uuencode, and x-uue are non-standard. If a "mac-binhex40" Content-Transfer-Encoding value is encountered, MIME readers SHOULD treat the MIME part body as if it had a Content-Type header value of "application/mac-binhex40" and process it as specified in section 2.2.4.2.3.

However, in the unlikely case of an actual "application/mac-binhex40" Content-Type, MIME readers SHOULD extract only the data fork from the MIME element content and use it as the value of the Attachment object's PidTagAttachDataBinary property. For X-uuencode and X-uue values, MIME readers SHOULD treat the attachment content as encoded with uuencode (see [IEEE1003.1]). The decoded value SHOULD be written to the Attachment object's PidTagAttachDataBinary property value.

### 2.2.4.2 Apple File Formats

[RFC1740] and [RFC1741] specify the use of the MIME Content-Types multipart/appledouble, application/applefile, and application/mac-binhex40 to encode files that originate from a Macintosh operating system, to preserve additional data that might be available for these files in that operating system. MIME readers SHOULD preserve this additional data for attached files to enable full support of Macintosh-based client applications.

In particular, the Attachment object content that is stored in PidTagAttachDataBinary MUST contain a MacBinary stream. This stream format incorporates both the resource and data forks, as well as certain metadata.

Note that MIME analysis of application/applefile attachments is specified differently, depending on whether the application/applefile MIME entity is a sub-part of multipart/appledouble.

#### 2.2.4.2.1 Multipart/Appledouble

A MIME element with Content-Type multipart/appledouble, as specified in [RFC1740], has two child MIME elements. The "header part" has a Content-Type of application/applefile; the "data part" can have any MIME content-type except application/applefile or another multipart Content-Type.

As a MIME reader copies data from a multipart/appledouble MIME entity to an Attachment object, it analyzes the three parts in the following sequence:

1. The header part (typically the first child of multipart/appledouble).

2. The data part (typically the second child of multipart/appledouble).

3. The multipart/appledouble envelope itself.

Property values that are set as a result of MIME header analysis of a particular MIME part, as specified in section 2.2.4.1, overwrite property values that are set as a result of previous MIME part analysis.

The procedure of header analysis for any part of a multipart/appledouble MIME part is similar to the procedure for ordinary file attachments specified in section 2.2.4.1, with the following additions:

MIME readers set the value of the PidTagAttachMimeTag property to "multipart/appledouble".

MIME readers set the value of the PidTagAttachEncoding property to the following byte sequence (in hexadecimal): "%x2A.86.48.86.F7.14.03.0B.01".

MIME readers copy the value of the Content-Type header on the data part to the value of the PidNameAttachmentMacContentType property. <175>

MIME readers SHOULD copy the entire MIME body of the header part to the value of the Attachment object's PidNameAttachmentMacInfo property.<176> MIME readers SHOULD also parse this data as an AppleSingle structure, as specified in [RFC1740], and combine it with the MIME body from the data part to form a MacBinary structure, which SHOULD then be written to the Attachment object's PidTagAttachDataBinary property.

MIME readers copy file creator and file type information taken from the MacBinary representation of the attachment, to the value of the PidTagAttachAdditionalInformation property, with special formatting as follows; the file creator and type fields are both unsigned 32-bit integers in big-endian format:

A single byte, value "0x3A" (colon character).

The file creator, encoded by the rule that follows.

A single byte, value "0x3A" (colon character).

The file type, encoded by the rule that follows.

A single byte, value "0x00".

Encoding is done from the highest-order byte to the lowest-order byte, by using the following scheme:

- Single bytes with values for "\" (%x5C), ":" (%x3A), and ";" (%x3B) are replaced with two-byte sequences: "\\" (%x5C.5C), "\:" (%x5C.3A), and "\;" (%x5C.3B) respectively.
- Single bytes with values less than 32, greater than 251, or equal to 127 are encoded by a backslash (%x5C), followed by the byte value in octal, padded with zeroes to 3 digits. So, for example, a "0x01" byte is encoded as "\001", and "0xFF" is encoded as "\377".

If parsing of the header part fails, MIME readers SHOULD reject the entire message as not MIME compliant.<177>

If the AppleSingle structure from the header part contains a file name for this attachment, it SHOULD be used as the file name only if no file name was found during processing of the MIME headers.<178>

#### 2.2.4.2.2 Application/Applefile

This section specifies MIME analysis for MIME parts with Content-Type application/applefile which are not sub-parts of a MIME part with Content-Type multipart/appledouble.

The procedure of MIME header analysis for application/applefile attachments is the same as for the procedure for ordinary file attachments specified in section 2.2.4.1, with one exception:

- MIME readers set the value of the PidTagAttachMimeTag property to "application/applefile".

Processing of MIME content SHOULD include parsing the AppleSingle structure, defined in [RFC1740]. MIME readers SHOULD use the data from this structure to fill the PidTagAttachDataBinary property and the PidNameAttachmentMacInfo property with appropriate structures, as specified in section 2.2.4.2.1.

If MIME body data does not match the definition of the AppleSingle structure (see [RFC1740]), MIME readers can choose to try to interpret the body of this MIME part as a MacBinary structure. If this succeeds, MIME readers SHOULD copy the resulting MacBinary structure to the value of the PidTagAttachDataBinary property, and PidNameAttachmentMacInfo SHOULD be filled with appropriate data from the MacBinary structure. The value of the PidNameAttachmentMacInfo property SHOULD be application/applefile data that contains only the header and resource fork sections. But if the MIME reader fails to parse the MIME body, the entire message SHOULD be rejected as not MIME compliant.<179>

If the AppleSingle or MacBinary structure contains a file name for this attachment, it SHOULD be used only if no file name was found during analysis of the attachment's MIME headers.<180>

The remainder of this section specifies how MIME readers SHOULD map elements from AppleSingle format, which can have Content-Type of multipart/appledouble, or application/applefile, to MacBinary data in the value of the PidTagAttachDataBinary property.<181>

The general structure of AppleSingle format is specified in [RFC1740]. In short, this data structure contains a header part, followed by some number of entries. Each of these entries is identified by a number (AppleSingleEntryId, unsigned 32-bit integer), which defines the internal structure of its binary data. The value of each AppleSingleEntryId, along with the definition of the structure of each entry, is specified by [RFC1740]. Custom entries are also allowed in this format.

The MacBinary structure consists of the following five parts; each part is padded to a 128-byte boundary, and all parts except the header are optional:<182>

1. Header
2. Additional header data

3. Actual file data (data fork)

4. Resource fork

5. Comment

The structure of the MacBinary header, with comments on usage of each field by MIME readers, is shown in the following table. All offsets and lengths are in bytes, and all integers use big-endian byte ordering.

| Field offset | Field length | Description |
|---|---|---|
| 0 | 1 | Old version number, MUST be zero. |
| 1 | 1 | Length of file name, unsigned byte; MUST be less than 64. |
| 2 | 63 | File name, in ASCII; characters beyond the length specified in byte 1 MUST be ignored.<183> |
| 65 | 4 | File type data, normally expressed as four characters. |
| 69 | 4 | File creator data, normally expressed as four characters. |
| 73 | 1 | Finder **flags**, bits 15:8. |
| 74 | 1 | Pad, MUST be 0 (zero). |
| 75 | 2 | Icon vertical location, unsigned 16-bit integer. |
| 77 | 2 | Icon horizontal location, unsigned 16-bit integer. |
| 79 | 2 | File's folder ID. |
| 81 | 1 | File protected flag, low order bit. |
| 82 | 1 | Pad, MUST be 0. |
| 83 | 4 | Data fork length, signed 32-bit integer, zero if there is no data fork. |
| 87 | 4 | Resource fork length, signed 32-bit integer, zero if there is no resource fork. |
| 91 | 4 | File creation date, signed 32-bit integer representing a number of seconds since (or before, if negative) midnight, 01/01/2000, UTC. |
| 95 | 4 | File modification date, signed 32-bit integer representing a number of seconds since (or before, if negative) midnight, 01/01/2000, UTC. |
| 99 | 2 | Comment length, unsigned 16-bit integer, MUST be 0 (zero). |
| 101 | 1 | Finder flags, bits 7:0. |
| 102 | 4 | Signature. MIME reader MUST set this value to "mBIN" (%x6D.42.49.4E).<184> |
| 106 | 1 | File name script **identifier**. MIME reader SHOULD set to 0 (zero). |
| 107 | 1 | Extended finder flags, MIME reader SHOULD set to 0 (zero). |
| 108 | 12 | Zero fill. |

| Field offset | Field length | Description |
|---|---|---|
| 120 | 2 | Secondary header length, MUST be 0 (zero). |
| 122 | 1 | MacBinary version number. MUST be set to 130 (0x82), indicating MacBinary III, when the MIME reader creates the MacBinary structure. |
| 123 | 1 | Minimum MacBinary version supported by this structure. MUST be set to 129 (0x81), indicating MacBinary II, when the MIME reader creates the MacBinary structure. |
| 124 | 2 | CRC of previous 124 bytes. MIME writers SHOULD NOT validate this value. MIME readers SHOULD calculate this value by appling a Cyclic Redundancy Check algorithm on the first 124 bytes of the header. The CRC algorithm used by MacBinary is the CCITT algorithm, which uses the polynomial 0x1024. For more information on CRC-CCITT, see [X25]. |
| Bytes 126:127 | 2 | Zero fill. |

When processing AppleSingle data, MIME readers MUST map AppleSingle fields to MacBinary fields as specified in the following table.

| AppleSingleEntryId and type | MacBinary field | Comment |
|---|---|---|
| 1, data fork | Bytes 83:86 – length;<br>MacBinary data fork part | This mapping SHOULD only be used by MIME readers in MIME analysis of a standalone application/applefile, because (according to [RFC1740]) data fork SHOULD be in a separate MIME part in multipart/appledouble case. |
| 2, Resource fork | Bytes 87:90 – length;<br>MacBinary resource fork part | If length is 0 (zero), MIME writers SHOULD NOT create this entry in AppleSingle. |
| 3, ASCII string | Byte 1 – length,<br>Bytes 2:64 – ASCII string value (only length bytes used) | File name. Note that MacBinary limits this string to 63 bytes. Excess bytes MUST be truncated. |
| 8, ASFileDates structure, create | Bytes 91:94 | File creation date, MIME readers SHOULD map it for AppleSingle to MacBinary conversion. |
| 8, ASFileDates structure, modify | Bytes 95:98 | File modification date, MIME readers SHOULD map it for AppleSingle to MacBinary conversion. |
| 8, ASFileDates structure, access | None | MIME writers SHOULD set to 0 (zero) on conversion to AppleSingle. |
| 8, ASFileDates structure, backup | None | MIME writers SHOULD set to 0 (zero) on conversion to AppleSingle. |
| 9, ASFinderInfo structure, ioFlFndrInfo.fdType | Bytes 65:68 | File type information. |

| AppleSingleEntryId and type | MacBinary field | Comment |
|---|---|---|
| 9, ASFinderInfo structure, ioFlFndrInfo.fdCreator | Bytes 69:72 | File creator information. |
| 9, ASFinderInfo structure, ioFlFndrInfo.fdFlags | Byte 73 – bits 15:8<br>Byte 101 – bits 7:0 | File finder flags word. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion. |
| 9, ASFinderInfo structure ioFlFndrInfo.fdLocation.v | Bytes 75:76<185> | Icon vertical location MIME readers SHOULD map this element for AppleSingle to MacBinary conversion. |
| 9, ASFinderInfo structure, ioFlFndrInfo.fdLocation.h | Bytes 77:78<186> | Icon horizontal location. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion. |
| 9, ASFinderInfo structure, ioFlFndrInfo.fdFldr | Bytes79:80 | File **folder ID**. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion. |
| 9, ASFinderInfo structure, ioFlXFndrInfo | None | MIME writers SHOULD fill with zeros on conversion to AppleSingle. |
| 10, ASMacInfo structure, filler | None | MIME writers SHOULD fill with zeros on conversion to AppleSingle. |
| 10, ASMacInfo structure, ioFlAttrib, bit 1 | Byte 81, low order bit | Protected flag. MIME readers SHOULD map this element for AppleSingle to MacBinary conversion.<187> |

Conversion from a full AppleSingle structure, found in a standalone application/applefile MIME element that is not a child of multipart/appledouble, to a reduced AppleSingle structure that SHOULD be used as a child of multipart/appledouble, is done simply by removing the entry with AppleSingleEntryId equal to 1 (the data fork) and adjusting the AppleSingle header accordingly.

#### 2.2.4.2.3 Application/Mac-binhex40

This section specifies MIME analysis for MIME parts with Content-Type application/mac-binhex40, as specified in [RFC1741].

The procedure of MIME header analysis for application/mac-binhex40 attachments is the same as for the procedure for ordinary file attachments that is specified in section 2.2.4.1, with the following exceptions:

1. MIME readers set the value of the PidTagAttachMimeTag property to "application/mac-binhex40".
2. The value of the Content-Transfer-Encoding header MUST be ignored.<188> MIME readers use BinHex decoding, as specified in [RFC1741], instead.

Processing of the MIME body SHOULD include parsing a binary structure of the decoded content, as specified in [RFC1741]. MIME readers SHOULD use the header and resource fork data from this structure to fill the PidNameAttachmentMacInfo property with appropriate data, as specified in section 2.2.4.2.1. MIME readers SHOULD also use this data to fill the MacBinary structure, which SHOULD be written to the value of the PidTagAttachDataBinary property.<189>

If parsing of the BinHex data fails, the entire message SHOULD be rejected by the MIME reader as not MIME compliant.<190>

MIME readers SHOULD copy the attachment file name that is extracted from the BinHex structure to the value of PidTagAttachFilename, but only if no file name was found during analysis of the MIME headers.<191><192>

### 2.2.4.3 Attached Messages

If an attachment MIME part has its Content-Type MIME header set to message/rfc822 (or no Content-Type header is present, and this MIME part is a sub-part of the multipart/digest MIME part), MIME readers SHOULD treat this attachment as an embedded message attachment, and set the value of the Attachment object's PidTagAttachMethod property to "5".

MIME analysis for MIME headers SHOULD be performed by the server in the same way as for ordinary file attachments, with the exception that the procedure for extracting the display name and file name for the attachment is different.

The display name for embedded message attachments is extracted from MIME part headers in the following order:

1. If a Content-Type MIME header is available on the attachment MIME part, and a non-empty name parameter is available on this header, its value SHOULD be used.
2. Otherwise, if a Content-Disposition MIME header is available on the attachment MIME part, and a non-empty filename parameter is available on this header, its value SHOULD be used.
3. Otherwise, if a Content-Description MIME header is available on the attachment, and its value is non-empty, it SHOULD be used.<193>
4. Otherwise, if a Subject header is available on the attachments, and its value is non-empty, it SHOULD be used.
5. If none of these conditions apply, the MIME reader SHOULD generate a name at random, or use a name derived from some other text value related to the Attachment object.

The resulting value SHOULD be written to the PidTagDisplayName property on the attachment, and then processed further to obtain a valid file name, as follows:

1. All Unicode separator characters in the file name SHOULD be replaced with the "?" character (U+003F).
2. All trailing and starting space and "." characters SHOULD be removed.

The file name is then separated into base and extension parts. To do this, the server SHOULD look for the last occurrence of any of the following characters:

- backslash, "\", U+005C
- forward slash, "/", U+002F
- colon, ":", U+003A
- period, ".", U+002E

If a "." (U+002E) character is the last one found, the part of the file name that precedes this character is considered to be base, and the rest is considered to be extension. In all other cases, extension is considered to be an empty string, and base part is considered to be the same as whole file name.

The resulting file name value SHOULD be written to the PidTagAttachLongFilename property, and the resulting extension value SHOULD be saved in the PidTagAttachExtension property.<194> <195> The file name SHOULD then be processed further to obtain a valid 8.3 name, as follows:

1. The value SHOULD be first separated into base and extension parts, by using the last "." character as a separator (if no such character is present, the extension is considered to be empty; the separator character itself is not included in the name or extension).

2. "+", ",", "=", "[", "]", and ";" characters SHOULD be replaced with the "_" (underscore) character.

3. Space, ".", "\", "/", "*", "<", ">", "?", ":", and "|" characters, as well as characters with UTF8 code greater than 127, SHOULD be removed.

4. If the base becomes an empty string, a non-empty string, such as a random string or other auto-generated value, SHOULD be used.

5. The base part of the file name SHOULD be trimmed to 8 characters; the extension part to 3 characters.

If either the name or the extension changed, the base part SHOULD additionally be trimmed to 6 characters, and "~1" SHOULD be added to its end.

The file name is saved in the <base>.<extension> format.

The resulting file name SHOULD be written to PidTagAttachFilename.<196>

The MIME part body for this attachment SHOULD be used for further MIME analysis that SHOULD result in assigning values to the properties of the embedded message from this attachment. This MIME analysis is performed in a way that is similar to that for ordinary MIME messages, with the following exceptions:

1. The X-MS-Exchange-Organization-Original-Sender MIME header value SHOULD be saved in the PidNameQuarantineOriginalSender property, if the header value is present.<197><198>

2. Unknown MIME headers, starting with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-", SHOULD NOT be excluded from analysis.<199><200>

After MIME analysis is done for the embedded message, the PidTagMessageFlags property SHOULD be modified; the mfUnsent flag ("0x8") SHOULD be removed, and the mfRead flag SHOULD be reset to 0x0.

#### 2.2.4.4 Inbound vCard Conversion

How the vCard format is converted to a Contact object when received by the messaging server is specified in [MS-OXVCARD]. This section describes additional information about converting an incoming vCard MIME part to its contact object counterpart.

vCard can appear in any multipart MIME part as well as the root content-type, as specified in [RFC2426]: The vCard format can be sent as the primary body or as the alternative body of a message. In message object format, however, vCard/contact information is only exposed as an attachment.

- When the vCard format is encountered at the root level of a MIME message, MIME reader promotes the vCard as a **contact attachment** to a message object with no body.

- When the vCard format is encountered within a multipart/alternative MIME part, MIME reader promotes the vCard content as a contact attachment and disregards it as a message body.

#### 2.2.4.4.1 Content-Type

The vCard MIME part is characterized by a Content-Type of "text/directory" with a profile of "vCard" for vCard format Version 3.0, as specified in [MS-OXVCARD]. If the Content-Type is "text/x-vCard", then the vCard MIME part uses vCard format Version 2.1, as specified in [RFC2426]. <201>

#### 2.2.4.4.2 General Parsing Guidelines

- If there are multiple instances of a property value that can be promoted to one single-valued **Messaging Application Programming Interface (MAPI)** property, then the last instance found is used, and others are dropped.
- vCard v2.1 allows parameters with the "type=" tag omitted. vCard v3.0 requires the "type=" tag.
- <grouping>.<property> is treated as <property>.
- This protocol handles both base64 encoding and quoted-printable encoding.
- Unknown properties will be dropped.

### 2.2.5 External Body Attachments

Attachment MIME parts with a Content-Type MIME header set to "message/external-body" SHOULD be analyzed in the same way as ordinary file attachments, with the exceptions specified in this section.<202>

If the Content-Type MIME header has no access-type parameter, or if the value of that parameter is not "anon-ftp", MIME readers SHOULD save the entire MIME part in the PidTagAttachDataBinary property on the attachment.

Otherwise, the following differences in MIME analysis apply:

- Different file name extraction logic SHOULD be applied, as specified later in this section.
- MIME readers SHOULD ignore the MIME part body. Instead, a specially formatted URL data string is saved in the PidTagAttachDataBinary property in ASCII format, as specified in this section.
- In this case, MIME readers expect the name, site, directory, and mode parameters to be present in the Content-Type MIME header. Clients SHOULD NOT create MIME that does not meet this criteria.

The URL data string to save in the PidTagAttachDataBinary property is constructed as follows. The values xchar and lowalpha used in this definition are specified in [RFC1738].

```
"[Internet Shortcut]" CR LF "URL=ftp://" site "/" directory "/" name [mode]
;contains header "site" parameter value
site=1*xchar

;contains header "directory" parameter value
directory=1*xchar
;contains header "name" parameter value
name=1*xchar
;if header "mode" parameter is "ascii", contains ";type=a"
;if header "mode" parameter is "image, contains ";type=i"
```

```
;otherwise not present
mode=";type=" 1*lowalpha
```

The file name extraction logic is similar to that for ordinary file attachments, with the following exceptions:

1. MIME readers use the name parameter value from the Content-Type MIME header as a value of the attachment file name. The file extension (a part of the file name after the last appearance of the "." character) SHOULD be replaced with "URL"; if the original file name has no extension, "URL" SHOULD be added at the end of the file name string.<203>
2. The sanitizing logic specified in section 2.2.4.1.1 SHOULD NOT be applied in this case.<204>
3. The file name value constructed in Step 1 SHOULD be used as an attachment display name as well.
4. All additional filtering logic specified in section 2.2.4.1.1 still applies in this case.
5. The procedure for calculating a value for the PidTagAttachFilename property is the same as for embedded message attachments specified in section 2.2.4.3.<205>

### 2.2.6 Reading Pure MIME Messages

The MIME reader is responsible for reading MIME messages and saving them in a way that minimizes changes to the original MIME content. This process is specified in section 2.4.

After the MIME reader converts the MIME message to a message object as specified in section 2.2, the reader copies the structure of the MIME message plus the contents of any message parts that are not converted, into the PidTagMimeSkeleton property, specified in section 2.4.<206> The process for copying message parts into the structure of **PidTagMimeSkeleton** is specified in section 2.4.3.

## 2.3 Additional Content Types

### 2.3.1 Analysis of Non-MIME Content

Internet message content that lacks a MIME-Version header can still be supported by MIME readers. The absence of a MIME-Version header makes the payload of SMTP or elsewhere non-MIME, with different behavior for inline attachments; headers such as Content-Type, Content-Disposition, and such have no special meaning. MIME readers can, nevertheless, assume the presence of a MIME-Version field and treat headers such as Content-Type as specified in [RFC2045] and elsewhere in this document.

The following is an example of such a message.

```
From: <user1@example.com>
To: <user2@example.com>
Subject: Example Legacy 822 message with attachment.
Date: Mon, 10 Mar 2008 14:36:46 -0700

this is a test message

begin 664 Flag.png
MB5!.1PT*&@H````-24A$4@```!0````-"`8```"I4$Y>````!F)+1T0`_P#_
M`/^@O:>3````"7!(67,```L3```+$P$`FIP8````!W1)344'V`,+`!8G&XK)
```

```
MCP```1M)1$%4.,NMDTUJPE`4A;\7\Z,ET8$-07`@BE,1.Q)T#2*X!-V+6^@J
M7(2.Q14X<J+6@29$DZBOHTJA)GW0GO&Y'^=>SA6\O4O^49J*R;4#7#OX.W#2
MGU,I'6EZ6YK>EDKIR*0_SP2*M)6=_(6"D1!$%F%L`O!BQMA6Q#DQ\"]Y]82M
MZH919\G.=QXP@#`VV?D.H\Z25G6CGM#(W4ANN<S5TCP_$FI",AW.L/1K*LS2
MKTR',S0AU1+FM#NW>W8!TCQ/IQKNGG%OD0H;]Q8TW+WZ#0M&@A`2VXKX"&SN
M4CS.\6H'!)&%E()S8J@E/"<&86PR:*^HE0]TZVNZ]36U\H%!>T48FT]AF3W\
F+J]X`F![*O[Z*;K*.ZF`OO0)9G1H4-$`H@T`````245.1*Y"8((`
`
end

begin 664 Flag.png
MB5!.1PT*&@H````-24A$4@```!0````-"`8```"I4$Y>````!F)+1T0`_P#_
M`/^@O:>3````"7!(67,```L3```+$P$`FIP8````!W1)344'V`,+`!8G&XK)
MCP```1M)1$%4.,NMDTUJPE`4A;\7\Z,ET8$-07`@BE,1.Q)T#2*X!-V+6^@J
M7(2.Q14X<J+6@29$DZBOHTJA)GW0GO&Y'^=>SA6\O4O^49J*R;4#7#OX.W#2
MGU,I'6EZ6YK>EDKIR*0_SP2*M)6=_(6"D1!$%F%L`O!BQMA6Q#DQ\"]Y]82M
MZH919\G.=QXP@#`VV?D.H\Z25G6CGM#(W4ANN<S5TCP_$FI",AW.L/1K*LS2
MKTR',S0AU1+FM#NW>W8!TCQ/IQKNGG%OD0H;]Q8TW+WZ#0M&@A`2VXKX"&SN
M4CS.\6H'!)&%E()S8J@E/"<&86PR:*^HE0]TZVNZ]36U\H%!>T48FT]AF3W\
F+J]X`F![*O[Z*;K*.ZF`OO0)9G1H4-$`H@T`````245.1*Y"8((`
`
end
```

MIME writers SHOULD NOT generate messages in this format; MIME SHOULD be generated instead. MIME readers SHOULD analyze messages in this format into headers, plain text body, and attached files (possibly including a TNEF attachment).

### 2.3.2 Message/Partial

The message/partial content type is not supported. MIME readers MUST NOT reassemble the individual messages of a message/partial message into a single message. MIME readers SHOULD<207> additionally reject messages that contain MIME entities with a Content-Type header of message/partial. This is to prevent virus scanning from being defeated by splitting up attachment content.

### 2.3.3 Multipart/Digest

This content type is treated exactly as multipart/mixed, except that the assumed Content-Type for body parts with no Content-Type header SHOULD be message/rfc822 rather than text/plain.

## 2.4 Preserving Unconverted MIME Parts on MIME Messages

The scenario where a server receives a pure MIME message using a MIME protocol (such as SMTP), and then must send it as a pure MIME message using a different MIME protocol (such as POP or IMAP), requires special functionality. The following diagram describes this process of "round-tripping."

SMTP => MIME reader => storage => MIME writer => POP or IMAP

On receipt, the MIME reader converts only selected portions of a pure MIME message to a Message object. In order for the MIME message to be round-tripped using the above scenario, a structure is needed that stores unconverted MIME parts in a way that minimizes changes to the original MIME content. <208>The structure used in this protocol is the PidTagMimeSkeleton property, specified in section 2.4.1, which is generated by the MIME reader during the MIME analysis process.<209>

After the MIME reader converts the MIME message to a Message object, as specified in section 2.2, the reader copies the structure of the MIME message plus the contents of any message parts that are not converted into the PidTagMimeSkeleton property on the Message object.

### 2.4.1 PidTagMimeSkeleton

This **PtypBinary** property contains all of the top level MIME message headers, all MIME message body part headers, and body part content that is not already converted to Message object properties, including attachments. Also included in PidTagMimeSkeleton are the contents of body parts that cannot be reliably reconstructed from properties, like iCalendar and vCard.

The following example illustrates how the contents of a MIME message are saved in PidTagMimeSkeleton and in Message properties.

### 2.4.2 Impact of Message Changes on the MIME Skeleton

When a stored message is modified through the RPC protocol, PidTagMimeSkeleton is deleted or left unchanged, depending on the changes made to the message:

1. Changes made to the following Message object properties SHOULD leave PidTagMimeSkeleton unchanged:
    1. PidTagIconIndex.
    2. **Reminders** on messages of type "IPM.NOTE".
    3. PidNameKeywords: which are mapped to the "Keywords" header in a MIME message.
2. Changes made to the following Message object properties leave PidTagMimeSkeleton unchanged; however, MIME writers SHOULD prefer the values of these properties over data from the MIME skeleton when generating MIME:
    1. Follow-up flags: These are stored in the PidLidFlagRequest property, which maps to the X-MessageFlag header in MIME. If this header already exists in PidTagMimeSkeleton, then its value SHOULD be updated when generating a message. If the X-MessageFlag header is updated, then it SHOULD be added to the top-level headers of a generated MIME message as well as to PidTagMimeSkeleton.
    2. PidTagImportance: The importance property maps directly to the Importance MIME header. If the client allows the value of this property on a message to be modified, then the importance header should be included in a generated MIME message and the change included in the stored PidTagMimeSkeleton.
3. Changes to any other Message object properties that affect the MIME structure or content of a message SHOULD result in deletion of the PidTagMimeSkeleton property associated with the message. The following table details the list of other Message object properties that fall into this category.

| Property name | Property name | Property name |
|---|---|---|
| PidLidClassification | PidTagAttachExtension | PidTagMessageClass |
| PidLidClassificationDescription | PidTagAttachFilename | PidTagMessageCodepage |
| PidLidClassificationGuid | PidTagAttachFlags | PidTagMessageDeliveryTime |
| PidLidClassificationKeep | PidTagAttachLongFilename | PidTagMessageFlags |
| PidLidClassified | PidTagAttachMethod | PidTagMessageLocaleId |
| PidLidInfoPathFormName | PidTagAttachMimeTag | PidTagNormalizedSubject |
| PidLidPercentComplete | PidTagAttachPayloadClass | PidTagObjectType |
| PidLidTaskComplete | PidTagAttachPayloadProviderGuidString | PidTagOriginatorDeliveryReportRequested |
| PidLidTaskDateCompleted | PidTagAttachSize | PidTagPriority |
| PidLidTaskDueDate | PidTagAutoForwarded | PidTagPurportedSenderDomain |

| Property name | Property name | Property name |
|---|---|---|
| PidLidTaskStartDate | PidTagAutoResponseSuppress | PidTagReadReceiptRequested |
| PidLidTaskStatus | PidTagBody | PidTagRecipientType |
| PidLidToDoTitle | PidTagBodyContentId | PidTagRenderingPosition |
| PidNameAcceptLanguage | PidTagBodyContentLocation | PidTagReplyRecipientEntries |
| PidNameAttachmentMacContentType | PidTagCallId | PidTagReplyRecipientNames |
| PidNameAttachmentMacInfo | PidTagClientSubmitTime | PidTagReplyTime |
| PidNameContentBase | PidTagContentFilterSpamConfidenceLevel | PidTagRtfCompressed |
| PidNameContentClass | PidTagConversationIndex | PidTagSenderIdStatus |
| PidNameContentType | PidTagConversationTopic | PidTagSenderName |
| PidNameCrossReference | PidTagCreationTime | PidTagSenderTelephoneNumber |
| PidNameKeywords | PidTagDisplayName | PidTagSendInternetEncoding |
| PidNameQuarantineOriginalSender | PidTagEmailAddress | PidTagSendRichInfo |
| PidNameXCallId | PidTagEntryId | PidTagSensitivity |
| PidNameXFaxNumberOfPages | PidTagExpiryTime | PidTagSentRepresentingAddressType |
| PidNameXSenderTelephoneNumber | PidTagFaxNumberOfPages | PidTagSentRepresentingEmailAddress |
| PidNameXVoiceMessageAttachmentOrder | PidTagFlagCompleteTime | PidTagSentRepresentingEntryId |
| PidNameXVoiceMessageDuration | PidTagFlagStatus | PidTagSentRepresentingName |
| PidNameXVoiceMessageSenderName | PidTagHtml | PidTagSmtpAddress |
| PidTagAddressBookProxyAddresses | PidTagIconIndex | PidTagSubject |
| PidTagAddressType | PidTagInReplyToId | PidTagSubjectPrefix |
| PidTagAttachAdditionalInformation | PidTagInternetCodepage | PidTagTextAttachmentCharset |
| PidTagAttachContentBase | PidTagInternetMessageId | PidTagTnefCorrelationKey |
| PidTagAttachContentId | PidTagInternetReferences | PidTagToDoItemFlags |
| PidTagAttachContentLocation | PidTagLastModificationTime | PidTagTransportMessageHeaders |
| PidTagAttachDataBinary | PidTagListHelp | PidTagVoiceMessageAttachmentOrder |
| PidTagAttachDataObject | PidTagListSubscribe | PidTagVoiceMessageDuration |

| Property name | Property name | Property name |
|---|---|---|
| PidTagAttachEncoding | PidTagListUnsubscribe | PidTagVoiceMessageSenderName |
| PidTagSentRepresentingEmailAddress | PidTagSessionInitiationProtocolUri | PidTagSenderSmtpAddress |
| PidTagReadReceiptName | PidTagReadReceiptEmailAddress | PidTagReadReceiptAddressType |
| PidNameXUnifiedMessagingPartnerContent | PidNameXUnifiedMessagingPartnerContext | PidNameXUnifiedMessagingPartnerStatus |
| PidNameXUnifiedMessagingPartnerAssignedId | PidNameAuthenticatedAs | PidNameAuthenticatedDomain |
| PidNameAuthenticatedMechanism | PidNameAuthenticatedSource | PidNameApprovalAllowedDecisionMakers |
| PidNameApprovalRequestor | PidNameRightsProtectMessage | PidNameOriginalSpamConfidenceLevel |
| PidTagContentFilterPhishingConfidenceLevel | PidLidInboundICalStream | PidNameLocationUrl |
| PidLidSingleBodyICal | PidTagReportingMessageTransferAgent | PidTagOriginalMessageId |
| PidTagRemoteMessageTransferAgent | PidNameOutlookProtectionRuleVersion | PidNameOutlookProtectionRuleTimestamp |
| PidNameOutlookProtectionRuleOverridden | | |

### 2.4.3 MIME Conversion

During MIME conversion, MIME reader keeps track of which MIME parts are saved as attachments, and which MIME part is promoted as the message body. If any of these MIME parts is missing a Content-Id header, MIME reader generates an X-Exchange-MIME-Skeleton-Content-Id header and promotes it to PidTagBodyContentId or PidTagAttachContentId as appropriate. If the message did not have a TNEF MIME part, or it was not promoted, after the message is converted, the whole MIME message SHOULD be saved in PidTagMimeSkeleton, with the MIME part content filtered out for MIME parts that were promoted as a message attachment or message body.

Body part contents for the following content-types are included in PidTagMimeSkeleton even when MIME reader has copied their contents to the Message object.

1. MIME part content for vCard attachments is not removed from PidTagMimeSkeleton, even though the content is converted to a Contact object.
2. Calendar message HTML body part content is not filtered out from PidTagMimeSkeleton, because conversion to the **Calendar object** is saved in a different form (RTF).
3. iCalendar MIME parts are not filtered out of PidTagMimeSkeleton.
4. For **S/MIME** messages, PidTagMimeSkeleton contains only root part headers since the rest of the data can be retrieved from the attachment.

5. For delivery status notification messages, an original message attachment is filtered out only if it was promoted as an attachment. Human-readable body content is always filtered. Report parts are not filtered out.

# 3 Structure Examples

This example shows a very simple e-mail message in both MIME and Message object formats. The following is the message in MIME format:

```
Received: from mailer01.example.com by mailer02.example.com
  with Microsoft SMTP Server; Mon, 11 Feb 2008 14:45:44 -0800
From: <user1@example.com>
To: <user2@example.com>; <user3@example.com>
Subject: test message
Date: Mon, 11 Feb 2008 14:45:32 -0800
Message-ID: <000001c86cff$cf0dd670$ae62379d@mail.example.com>
MIME-Version: 1.0
Content-Type: text/plain
Content-Transfer-Encoding: 7bit
Importance: normal
Priority: normal

this is a test message
```

The following table shows this simple message represented as a Message object. The message itself has several properties, and it contains recipients each with several properties of its own. The recipients are shown in the table that follows.

| Message Property | Type | Value |
|---|---|---|
| PidTagMessageDeliveryTime | PtypTime | %xF9.2D.82.D6.FF.6C.C8.01 |
| PidTagSentRepresentingName | PtypString | Test user 1 |
| PidTagSentRepresentingAddressType | PtypString | SMTP |
| PidTagSentRepresentingEmailAddress | PtypString | user1@example.com |
| PidTagSubject | PtypString | test message |
| PidTagClientSubmitTime | PtypTime | %x00.8E.AD.CE.FF.6C.C8.01 |
| PidTagInternetMessageId | PtypString | <000001c86cff$cf0dd670$ae62379d@mail.example.com> |
| PidTagImportance | PtypInteger32 | 1 |
| PidTagPriority | PtypInteger32 | 0 |
| PidTagBody | PtypString | this is a test message |
| PidTagInternetCodepage | PtypInteger32 | 28591 |
| PidTagObjectType | PtypInteger32 | 5 |

| Message Property | Type | Value |
|---|---|---|
| PidTagMessageFlags | PtypInteger32 | 0x23 |

| Row ID | Recipient property | Type | Value |
|---|---|---|---|
| 38714304 | PidTagDisplayName | PtypString | Test user 2 |
| 38714304 | PidTagAddressType | PtypString | EX |
| 38714304 | PidTagEmailAddress | PtypString | /O=Example1/OU= Administrative Group/CN=recipients/CN=user2 |
| 38714304 | PidTagSmtpAddress | PtypString | user2@example.com |
| 38714304 | PidTagRecipientType | PtypInteger32 | 1 |
| 38714304 | PidTagObjectType | PtypInteger32 | 6 |
| 38714305 | PidTagDisplayName | PtypString | Test user 3 |
| 38714305 | PidTagAddressType | PtypString | EX |
| 38714305 | PidTagEmailAddress | PtypString | /O=Example1/OU= Administrative Group/CN=recipients/CN=user3 |
| 38714305 | PidTagSmtpAddress | PtypString | user3@example.com |
| 38714305 | PidTagRecipientType | PtypInteger32 | 1 |
| 38714305 | PidTagObjectType | PtypInteger32 | 6 |

While obviously less compact than the MIME format, the Message object format makes strongly typed data available. Both client and server code can sort, find, and process messages according to specific criteria such as "all unread messages," "all messages tagged as Personal," or "all calendar items occurring in the week of 2/12/2008, sorted by start time."

## 3.1 MIME Examples

### 3.1.1 MIME Message Containing Inline and Non-Inline Attachments

The following example demonstrates a MIME-formatted message that contains both inline and non-inline **attachments**, as specified in section 2.1.4.

```
From: <john@contoso.com>
To: <imtiaz@contoso.com>
Subject: Example with inline and non-inline attachments.
Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="simple boundary 1"

--simple boundary 1
Content-Type: multipart/related; boundary="simple boundary 2"
```

```
--simple boundary 2
Content-Type: multipart/alternative; boundary="simple boundary 3"

--simple boundary 3
Content-Type: text/plain

...Text without inline reference...
--simple boundary 3
Content-Type: text/html

...Text with inline reference...
--simple boundary 3--
--simple boundary 2
Content-Type: image/png; name="inline.PNG"
Content-Transfer-Encoding: base64
Content-ID: <6583CF49B56F42FEA6A4A118F46F96FB@example.com>
Content-Disposition: inline; filename="Inline.png"

...Attachment data encoded with base64...
--simple boundary 2--

--simple boundary 1
Content-Type: image/png; name=" Attachment "
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="Attachment.png"

...Attachment data encoded with base64...
--simple boundary 1--
```

### 3.1.2 MIME Message Containing Only Inline Attachments

The following example demonstrates a MIME-formatted message that contains only inline **attachments**, as specified in section 2.1.4.

```
From: <john@contoso.com>
To: <imtiaz@contoso.com>
Subject: Example with inline attachment.
Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/related; boundary="simple boundary"

--simple boundary
Content-Type: text/html;

...Text with reference...

--simple boundary
Content-Type: image/png; name="inline.PNG"
Content-Transfer-Encoding: base64
Content-ID: <6583CF49B56F42FEA6A4A118F46F96FB@example.com>
Content-Disposition: inline; filename=" inline.png"

...Attachment data encoded with base64...
--simple boundary--
```

### 3.1.3 MIME Message Containing Only Non-Inline Attachments

The following example demonstrates a MIME-formatted message that contains only non-inline **attachments**, as specified in section 2.1.4.

```
From: <john@contoso.com>
To: <imtiaz@contoso.com>
Subject: Example with non-inline attachment.
Date: Mon, 10 Mar 2008 14:36:46 -0700
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="simple boundary"

--simple boundary
Content-Type: text/plain;

...Text without reference...

--simple boundary
Content-Type: image/png; name=" Attachment"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="Attachment.png"

...Attachment data encoded with base64...
--simple boundary--
```

# 4 Security Considerations

## 4.1 Unsolicited Commercial E-mail (Spam)

A significant business has evolved around the sending of unsolicited commercial e-mail (colloquially referred as spam). Unlike physical bulk mail, with its built-in **restrictions** on labeling and cost, the general structure of SMTP allows anonymous sources to send e-mail virtually without restriction. Attempts are being made to reduce the volume of spam that makes it to a person's Mailbox, but care has to be taken to not affect legitimate senders.

Part of the success of this industry is the fact that people impute importance to unverifiable things (see [MS-OXCSPAM]). For example, the purported sender of a piece of mail (considering most mail is not digitally signed) is commonly used by people to attach importance and priority. If the Message appears to come from a person's boss, there is a higher probability that the employee would act on the Message. In this case, care needs to be taken when receiving mail over unauthenticated transports. Even if the routing address of the sender matches a valid employee or contact, it is better if clients and servers preserve the external routing address on the Message object, because replacing it with its address book equivalent could impute elevated importance to the content.

## 4.2 Information Disclosure

Content that is sent can contain hints about the source network's topology and structure. In MIME, this can be discerned from the Received headers (every SMTP server and potentially the client's computer are listed by its network address). In addition, if the optional algorithm specified in [RFC2822] section 3.6.4 to generate a unique Message-Id header value is implemented, the client's network address and internal domain name is exposed. Alternately, a GUID can be used to satisfy the unique identifier, circumventing the first data exposure. The aforementioned problem also exists for Content-Id header and boundary parameter values. It is suggested that a GUID be used here as well.

Additionally, when sending recipient data other than the properties mentioned previously, implementations need to be aware that internal data can be exposed. For example, office numbers and phone numbers could be cached on each recipient. This is an issue on embedded messages that are transported via TNEF (see [MS-OXTNEF]), as TNEF has the ability to carry more recipient information than is available with MIME headers.

Care also needs to be taken when receiving mail to deal with some information disclosure issues. If the e-mail message leverages any feature that requires the client to "fetch" additional resources when displaying it, the act of fetching can expose the fact that the recipient is an actual employee, and the date and time that the message was read. Examples of this are external bodies, HTML stylesheets, and images.

## 4.3 Content-Type Versus File Extension Mismatch

Various clients accept the Content-Type received from the server, but then verify the content. This can include checking the file extension or looking for a thumbprint at the start of the file, and then mapping this data back to a verified Content-Type. If the file extension or thumbprint does not match the stated Content-Type, a Content-Type value derived from the file extension or thumbprint is used instead. This behavior is actually codified in [RFC2045], which allows the sender to set the Content-Type to "application/octet-stream" (or not set it at all). The recipient is then responsible for correctly determining the type of content via alternate means.

In addition, it was found that various clients incorrectly set the Content-Type either by mistake or intentionally. Support to address the former has existed for quite some time but has opened a path to potentially thwart policy scanning and protection applications running on the server.

Therefore MIME readers can correct mislabeled Content-Type header values so that server Policy Agents and clients can trust the header value. Clients need to offer a mechanism to do one or more of the following:

- Suppress correcting the Content-Type header.
- Block attachments by type or extension.
- Offer some sort of security barrier before running any script, or binary.

These steps are particularly important if the sender is unauthenticated.

## 4.4 Do Not Support Message/Partial

A Content-Type of "message/partial" allows large messages to be sent in pieces and re-assembled by the client. It was originally designed to work around transmission failures during slow delivery causing the complete message to be resent from scratch, and to work around message size restrictions of implementations of protocols like SMTP. With increased bandwidth speeds, and greater connectivity, the long transmission times are more a thing of the past. Continued support for this Content-Type allows an avenue for content that is inappropriate to reach (or leave) the e-mail client's computer. This could include things such as "Information disclosure" of proprietary information, unsolicited commercial e-mail (spam), and computer virus attachments.

E-mail servers attempt to protect their users from inappropriate content by implementing Policy applications that run as part of the protocol. For them to work efficiently, the complete content is incorporated into one message. For this reason, servers need to prohibit sending or receiving messages with a Content-Type of "message/partial".

## 4.5 Considerations for Message/External-Body

The original MIME RFC [RFC1521] allowed the body of an entity to be referenced externally rather than requiring it to be inline. The current MIME RFC [RFC2046] specifies the form of this construct; the security implications are as follows:

1. The blind retrieval of the content by the client can disclose information about the recipient.
2. The authentication mechanism tied to the retrieval (access-type parameter) can result in a pop-up dialog box, leading the user to expose credential information.
3. The server (Policy or delivery application) that is attempting to check the content opens up a denial of service vector for the remote host to tie up server resources.

## 4.6 Preventing Denial of Service Attacks

### 4.6.1 Submission Limits

Servers can limit the size of received messages to limit resource consumption. Such limits can be different for authenticated versus anonymous senders.

### 4.6.2 Complexity of Nested Entities

It is possible to represent very complex hierarchies with MIME and to add superfluous entity layers (multipart content-types). Servers and clients need to protect themselves from stack overflows or heap starvation. This can involve limiting the nesting depth of attachments and body parts within a single message.

### 4.6.3 Number of Embedded Messages

A server that implements this protocol converts MIME content into a Message object representation. This causes each embedded message to be mapped individually and all attachments to be included therein. One implementation of this is to recursively handle each attached embedded message, but care needs to be taken not to encounter a stack overflow by doing so.

### 4.6.4 Compressed Attachments

Analyzing each attachment on the server is a concern when decompression is required. It is possible to encounter compressed content that requires large volumes of disk space, memory, or other resources, leading to a denial of service.

# 5 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products:

- Microsoft® Office Outlook® 2003
- Microsoft® Exchange Server 2003
- Microsoft® Office Outlook® 2007
- Microsoft® Exchange Server 2007
- Microsoft® Outlook® 2010
- Microsoft® Exchange Server 2010
- Microsoft® Exchange Server 2010 Service Pack 1 (SP1)

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. 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 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 product does not follow the prescription.

<1> Section 2: Exchange 2010 does not support MIME data generation for the Content-Base, Content-Location, and Expires headers.

<2> Section 2.1: Office Outlook 2003 and Office Outlook 2007 do not use the HTML only setting.

<3> Section 2.1.1: Exchange 2003 only generates recipients' display names if the SMTP policy for the domain is configured to preserve sender's address on message, which is off by default.

<4> Section 2.1.1: Office Outlook 2003 and Office Outlook 2007 do not encapsulate addresses.

<5> Section 2.1.1.1: Office Outlook 2003 and Office Outlook 2007 do not generate MIME recipients for Bcc recipients if it is generating a message to send via SMTP.

<6> Section 2.1.1.1: Office Outlook 2003 and Office Outlook 2007 use the listed steps in the following order: 4, 5, 1, 2, 3, 7. Office Outlook 2003 and Office Outlook 2007 do not use step 6.

<7> Section 2.1.1.1.1: Exchange 2003 generates attRecipTable for top-level summary and legacy TNEF messages.

<8> Section 2.1.1.2: Office Outlook 2003 and Office Outlook 2007 do not copy the values of PidTagReplyRecipientEntries and PidTagReplyRecipientNames to the TNEF body part.

<9> Section 2.1.1.3: Office Outlook 2003 and Office Outlook 2007 do not copy the values of the PidTagSentRepresenting property group to the TNEF body part.

<10> Section 2.1.1.4: Exchange 2003 does not generate a Sender header.

<11> Section 2.1.1.4: Office Outlook 2003 and Office Outlook 2007 do not copy the values of the PidTagSender property group to the TNEF body part.

<12> Section 2.1.1.5: Exchange 2007 copies the values in the **PidTagSentRepresenting** property group to the Return-Receipt-To header.

<13> Section 2.1.1.5: Office Outlook 2003 and Office Outlook 2007 do not copy the values of PidTagOriginatorDeliveryReportRequested or the specified e-mail **properties** to the TNEF body part.

<14> Section 2.1.1.6: Office Outlook 2003 and Office Outlook 2007 do not copy the values of the PidTagReadReceipt and PidTagSentRepresenting groups of properties. Note that they do copy PidTagReadReceiptRequested.

<15> Section 2.1.1.8: Office Outlook 2003 and Office Outlook 2007 do not encapsulate addresses.

<16> Section 2.1.1.8: Exchange 2007 encodes only the address part and uses the address-type as-is. On decoding, Exchange 2007 scans for the first hyphen ("-") after "IMCE" and uses the prefix as-is without parsing for any escaped characters. Exchange 2007 has no limitation on the length of address-type. Exchange 2007 does not properly de-encapsulate if address-type contains an ASCII hyphen ("-").

<17> Section 2.1.1.8: Exchange 2003 builds the entire string including address-type and address, and then encodes the whole string, escaping any non-alphanumeric characters contained in address-type. On decoding, Exchange 2003 unescapes the entire string and scans the first nine characters for a hyphen ("-"). If a hyphen is not found, then the address is not de-encapsulated. The address-type is limited to eight characters. Only ASCII alphanumeric characters are allowed in address-type. Exchange 2003 does not properly de-encapsulate if address-type contains an ASCII hyphen ("-").

<18> Section 2.1.2: Office Outlook 2003 and Office Outlook 2007 do not copy properties to the TNEF body part if there is a corresponding MIME header.

<19> Section 2.1.2.1: Office Outlook 2003 and Office Outlook 2007 do not convert **appointment** items to MIME.

<20> Section 2.1.2.2: Exchange 2003 does not generate the Content-class header.

<21> Section 2.1.2.2: When the PidTagMessageClass value is "IPM.Note.Microsoft.Fax.CA", Exchange 2007 writes the Content-class header value as "fax".

<22> Section 2.1.2.2: When the PidTagMessageClass value is "IPM.Note.Microsoft.Voicemail.UM.CA", Exchange 2007 writes the Content-class header value as "voice".

<23> Section 2.1.2.3: Office Outlook 2003 does not support unified messaging.

<24> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-CallingTelephoneNumber header.

<25> Section 2.1.2.3: Exchange 2003 does not generate the X-CallingTelephoneNumber header.

<26> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-VoiceMessageDuration header.

<27> Section 2.1.2.3: Exchange 2003 does not generate the X-VoiceMessageDuration header.

<28> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-VoiceMessageSenderName header.

<29> Section 2.1.2.3: Exchange 2003 does not generate the X-VoiceMessageSenderName header.

<30> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-FaxNumberOfPages header.

<31> Section 2.1.2.3: Exchange 2003 does not generate the X-FaxNumberOfPages header.

<32> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-AttachmentOrder header.

<33> Section 2.1.2.3: Exchange 2003 does not generate the X-AttachmentOrder header.

<34> Section 2.1.2.3: Office Outlook 2003 and Office Outlook 2007 do not generate the X-CallID header.

<35> Section 2.1.2.3: Exchange 2003 does not generate the X-CallID header.

<36> Section 2.1.2.4: Exchange 2003 only implements encoding for three specific headers: "Subject", "Thread-Topic", and "X-Message-Flag", which is not fully compliant with [RFC2047].

<37> Section 2.1.2.4: Exchange 2003 does not check to prevent the use of the reserved name headers "X-Microsoft-Exchange-Organization" and "X-Microsoft-Exchange-Forest".

<38> Section 2.1.2.7: Office Outlook 2003 and Office Outlook 2007 do not copy PidTagClientSubmitTime to the TNEF body part.

<39> Section 2.1.2.8: Office Outlook 2003 and Office Outlook 2007 instead copy PidTagSubject to the Subject header.

<40> Section 2.1.2.8: Office Outlook 2003 and Office Outlook 2007 do not copy the message subject to the TNEF body part.

<41> Section 2.1.2.11: Office Outlook 2003 and Office Outlook 2007 will generate a new Message-ID if it is generating a message to send via SMTP.

<42> Section 2.1.2.14: Exchange 2003 does not copy the value of the PidTagInReplyToId property to the In-Reply-To header.

<43> Section 2.1.2.16: Office Outlook 2003 and Office Outlook 2007 do not generate the Accept-Language header.

<44> Section 2.1.2.16: Exchange 2003 does not generate the Accept-Language header.

<45> Section 2.1.2.16: Exchange 2003 does not generate the Content-Language header.

<46> Section 2.1.2.17: Office Outlook 2003 and Office Outlook 2007 do not generate any classification headers.

<47> Section 2.1.2.17: Exchange 2003 does not generate any classification headers.

<48> Section 2.1.2.18: Exchange 2007 and Exchange 2010 only generate X-Payload-headers when they are properties on a message, which includes message attachments. These headers are not generated when they are properties of file attachments.

<49> Section 2.1.2.18: Office Outlook 2003, Office Outlook 2007, and Outlook 2010 do not generate the X-Payload-Provider-GUID header.

<50> Section 2.1.2.18: Outlook 2010 sets the PidTagAttachPayloadProviderGuidString property as a property of the attachment, not of the message.

<51> Section 2.1.2.18: Exchange 2003 does not copy the value of the PidTagAttachPayloadProviderGuidString property to the X-Payload-Provider-Guid header.

<52> Section 2.1.2.18: Office Outlook 2003, Office Outlook 2007, and Outlook 2010 do not generate the X-Payload-Class header, nor will Office Outlook 2003 generate a PidTagAttachPayloadClass property from the X-Payload-**class** header on incoming messages.

<53> Section 2.1.2.18: Outlook 2010 sets the PidTagAttachPayloadClass property on the attachment, not on the message.

<54> Section 2.1.2.18: Exchange 2003 does not copy the value of the PidTagAttachPayloadClass property to the X-Payload-Class header.

<55> Section 2.1.2.19: Office Outlook 2003 generates the X-MS-Has-Attach header with no output.

<56> Section 2.1.2.20: Office Outlook 2003 and Office Outlook 2007 do not generate the X-Auto-Response-Suppress header.

<57> Section 2.1.2.20: Exchange 2003 does not generate the X-Auto-Response-Suppress header.

<58> Section 2.1.2.21: Office Outlook 2003 and Office Outlook 2007 do not generate the X-MS-Exchange-Organization-AutoForwarded header.

<59> Section 2.1.2.21: Exchange 2003 does not generate the X-MS-Exchange-Organization-AutoForwarded header.

<60> Section 2.1.2.22: Office Outlook 2003 and Office Outlook 2007 do not generate the X-MS-Exchange-Organization-SenderIdResult header.

<61> Section 2.1.2.22: Exchange 2003 does not generate the X-MS-Exchange-Organization-SenderIdResult header.

<62> Section 2.1.2.23: Office Outlook 2003 and Office Outlook 2007 do not generate the X-MS-Exchange-Organization-PRD header.

<63> Section 2.1.2.23: Exchange 2003 does not generate the X-MS-Exchange-Organization-PRD header.

<64> Section 2.1.2.24: Office Outlook 2003 and Office Outlook 2007 do not generate the X-MS-Exchange-Organization-SCL header.

<65> Section 2.1.2.24: Exchange 2003 does not generate the X-MS-Exchange-Organization-SCL header.

<66> Section 2.1.2.28: Exchange 2003 does not write the Reply-By header.

<67> Section 2.1.3.1: Exchange 2003 defaults to ISO-8859-1 when PidTagInternetCodepage is not set by the client for plain text message bodies.

<68> Section 2.1.3.1: Office Outlook 2003 and Office Outlook 2007 either convert the RTF text to HTML or generate a TNEF attachment that contains the RTF body.

<69> Section 2.1.3.1: If a different charset is supplied for different body types, Exchange 2007 identifies the body format that can generate the others, and applies its charset to all body formats.

<70> Section 2.1.3.2: This is for compatibility with the Exchange 2003 TNEF Reader, which loses body text if only a plain text body is encoded in TNEF.

<71> Section 2.1.3.4: Exchange 2003 relies on **store** for text conversions, so it copies the value of PidTagBody.

<72> Section 2.1.3.6: Exchange 2003 and Exchange 2007 do not check to determine whether an apparently inline attachment is, in fact, referenced from the message body.

<73> Section 2.1.3.8: Exchange 2007 and Exchange 2010 support a configuration option to select a charset encoding other than UTF-8 for the MIME headers and MIME entities of a message object. The iCal message body, however, always remains encoded in UTF-8.

<74> Section 2.1.3.8.2: Exchange 2003 uses "quoted-printable" Content-Transfer-Encoding. Exchange 2007 uses base64 Content-Transfer-Encoding. Both encodings produce conformant MIME entities, and both encodings are consistent with the description in this section.

<75> Section 2.1.3.8.2: Exchange 2003 relies on the store for text conversions, so it copies the value of the PidTagHtml property.

<76> Section 2.1.3.8.2: Exchange 2003 uses Content-Transfer-Encoding: 8bit for the text/calendar MIME entity referred to in this section, whereas Exchange 2007 uses Content-Transfer-Encoding: base64. Both are conformant MIME, and both are consistent with the description in this section.

<77> Section 2.1.3.8.3: Exchange 2003 generates a MIME entity with "multipart/mixed" for the value of its Content-Type header.

<78> Section 2.1.4.1: Exchange 2007 does not exclude attached Message objects or attachments to plain text messages from being considered inline.

<79> Section 2.1.4.1.1: Exchange 2007 does not exclude non-OLE attachments in an RTF message from being considered inline.

<80> Section 2.1.4.1.2: Exchange 2003 does not generate the cid URI.

<81> Section 2.1.4.1.2: The MIME writer in Exchange 2007 only checks condition 1, not conditions 2 or 3, when classifying attachments as inline.

<82> Section 2.1.4.1.2: Exchange 2003 does not automatically generate the Content-Disposition header for inline attachments.

<83> Section 2.1.4.2.2: Exchange 2003 only generates PidTagAttachMimeTag on inbound MIME, and does not provide the property value for an outbound MIME message.

<84> Section 2.1.4.2.2: Office Outlook 2003 and Office Outlook 2007 do not generate the Content-Description header.

<85> Section 2.1.4.2.2: Office Outlook 2003 does not perform any encoding of the non-ASCII characters in the Content-Description header. It uses the attachment's PidTagDisplayName value with no further encoding.

<86> Section 2.1.4.2.2: Office Outlook 2003 does not generate the Content-Disposition header for inline attachments.

<87> Section 2.1.4.2.2: Office Outlook 2003 and Office Outlook 2007 do not generate the size parameter for the Content-Disposition header.

<88> Section 2.1.4.2.2: Exchange 2003 does not generate the size parameter for the Content-Disposition header.

<89> Section 2.1.4.2.2: Office Outlook 2003 and Office Outlook 2007 do not generate the creation-date parameter for the Content-Disposition header.

<90> Section 2.1.4.2.2: Exchange 2003 does not generate the creation-date parameter for the Content-Disposition header.

<91> Section 2.1.4.2.2: Exchange 2007 appends the word "GMT" to the creation-date parameter without adjusting the time from the server time zone to GMT.

<92> Section 2.1.4.2.2: Office Outlook 2003 and Office Outlook 2007 do not generate the modification-date parameter for the Content-Disposition header.

<93> Section 2.1.4.2.2: Exchange 2003 does not generate the modification-date parameter for the Content-Disposition header.

<94> Section 2.1.4.2.2: Exchange 2007 appends the word "GMT" to the modification-date parameter without adjusting the time from the server time zone to GMT.

<95> Section 2.1.4.3: Office Outlook 2003 and Office Outlook 2007 do not encode attachments in the MacBinary format.

<96> Section 2.1.4.3: Exchange 2007 does not ignore the presence of secondary header data in a MacBinary stream; the behavior of Exchange 2007 is undefined if secondary header data is present.

<97> Section 2.1.4.3: If bytes 120:121 have non-zero values, then Exchange 2003 writes the attachment content type as application/octet-stream instead of application/appledouble. The resource and data fork are not returned as separate attachments.

<98> Section 2.1.4.3: Exchange 2007 does not ignore the presence of additional data in a MacBinary stream; the behavior of Exchange 2007 is undefined if additional data is present.

<99> Section 2.1.4.3: If byte 99 has a nonzero value, then Exchange 2003 and Exchange 2007 write the attachment content type as application/octet-stream instead of application/appledouble. The resource and data fork are not returned as separate attachments.

<100> Section 2.1.4.3: Exchange 2003 uses a maximum field length of 62 bytes for the file name, rather than 63 bytes.

<101> Section 2.1.4.4: Exchange 2003 sets the Content-Type header to "image/BMP" and uses "ole#.BMP" as the file name, where "#" is a digit representing the index number of the attached file. Exchange 2007 converts OLE renderings to jpeg. Exchange 2003 keeps OLE renderings in their bitmap type.

<102> Section 2.1.4.4: Exchange 2003 does not produce a Content-Description header for OLE attachments.

<103> Section 2.1.4.4: Office Outlook 2003 and Office Outlook 2007 do not convert OLE attachments. OLE attachments are omitted from the MIME version of the message.

<104> Section 2.1.4.6: Exchange 2010 does not support vCard Version 2.1 on outbound MIME messages.

<105> Section 2.1.5: Exchange 2003 and Exchange 2007 do not support the functionality specified in this section.

<106> Section 2.1.5: Outlook 2010 does not use the PidTagMimeSkeleton property.

<107> Section 2.1.5.1: Exchange 2010 sets the MIME-Version header to "1.0".

<108> Section 2.2: Outlook 2010 preferences TNEF body part data, but will discard information from the TNEF (such as **recipients**) that it does not need.

<109> Section 2.2.1.1: Office Outlook 2003 and Office Outlook 2007 do not check for IMCEA encapsulation and do not perform de-encapsulation.

<110> Section 2.2.1.2: Office Outlook 2003 and Office Outlook 2007 do not check for IMCEA encapsulation and do not perform de-encapsulation.

<111> Section 2.2.1.3: Office Outlook 2003 and Office Outlook 2007 will not use the attSentFor attribute or the PidTagSentRepresenting value if the From header is absent.

<112> Section 2.2.2: Exchange 2007 uses the last **instance** to set the value of the corresponding property.

<113> Section 2.2.2.5: Exchange 2003 only generates the Importance header, not the X-Priority header.

<114> Section 2.2.2.5: Exchange 2003 maps the header "Priority: Urgent" to PidTagImportance with value 0x00000001.

<115> Section 2.2.2.7: Exchange 2007 sets the PidTagConversationTopic property differently depending on which MIME header, Subject header, or Thread-Topic header is present, the value of the header, and which header comes first. If neither header is available, then PidTagConversationTopic is not set. If only the Subject header is available and the header can be parsed to set the PidTagNormalizedSubject, then PidTagConversationTopic is set to PidTagNormalizedSubject; however, if the message class is "IPM.Post" (denoting a Post object as defined in [MS-OXOPOST]), then PidTagConversationTopic is not set. If only a Thread-Topic is available, then PidTagConversationTopic is set from the value of the Thread-Topic header. If both Subject and Thread-Topic are present, and the Subject header precedes Thread-Topic, then Exchange 2007 attempts to normalize the Subject value, as specified in section 2.2.2.6.1. If normalization is successful, then Exchange 2007 sets PidTagConversationTopic to PidTagNormalizedSubject. If unsuccessful, then Exchange 2007 looks to see whether the Subject header value ends with the Thread-Topic header value. If this is successful, then Exchange 2007 sets PidTagConversationTopic to the value of Thread-Topic. If neither of the above conditions are successful, then Exchange 2007 sets PidTagConversationTopic to the value of the Subject header. If both Subject and Thread-Topic are present, and Thread-Topic precedes Subject, then Exchange 2007 first looks to see whether the Subject header value ends with the Thread-Topic header value. If successful, then Exchange 2007 sets PidTagConversationTopic to the value of Thread-Topic. If unsuccessful, then Exchange 2007 normalizes the value of the Subject header, as specified in section 2.2.2.6.1, producing PidTagSubjectPrefix and PidTagNormalizedSubject, and sets PidTagConversationTopic to PidTagNormalizedSubject. If neither of the above conditions are successful, then Exchange 2007 sets PidTagConversationTopic to the value of the Subject header.

<116> Section 2.2.2.11: Office Outlook 2003 and Office Outlook 2007 do not copy the value of the Accept-Language or X-Accept-Language header.

<117> Section 2.2.2.11: Exchange 2003 does not copy the value of the Accept-Language or X-Accept-Language header.

<118> Section 2.2.2.13: Exchange 2007 uses whichever header shows up last in the list of header information, either the Expires header or the Expiry-Date header, as the header used to set PidTagExpiryTime.

<119> Section 2.2.2.14: Outlook 2010 does not read the X-Auto-Response-Suppress header.

<120> Section 2.2.2.14: Exchange 2003, Exchange 2007, and Exchange 2010 set the value of the PidTagAutoResponseSuppress property to 0x00000000 in this instance.

<121> Section 2.2.2.14: Office Outlook 2003 and Office Outlook 2007 ignore the X-AUTO-Response-Suppress and Precedence headers.

<122> Section 2.2.2.14: Exchange 2003 ignores the X-AUTO-Response-Suppress and Precedence headers.

<123> Section 2.2.2.15: Exchange 2003 does not generate the Content-class header.

<124> Section 2.2.2.15: Exchange 2003 sets the value of the PidTagMessageClass property for all Content-Class header values specified in section to "IPM.Note".

<125> Section 2.2.2.15: Office Outlook 2003 and Office Outlook 2007 do no special processing for "urn:content-class:custom."

<126> Section 2.2.2.15: Exchange 2003 does no special processing for "urn:content-class:custom."

<127> Section 2.2.2.16: Exchange 2003 does not support the id-mapped properties used in this section: PidLidToDoTitle, PidLidTaskStatus, PidLidTaskDueDate, PidLidTaskStartDate, PidLidTaskDateCompleted, PidLidTaskComplete, and PidLidPercentComplete.

<128> Section 2.2.2.17: Outlook 2010 does not read the X-List-Help, X-List-Subscribe, or X-List-Unsubscribe headers.

<129> Section 2.2.2.18: Exchange 2003 does not write an X-Payload-class and an X-Payload-Provider-GUID header.

<130> Section 2.2.2.18: Outlook 2010 does not read or write the X-Payload-Class and X-Payload-Provider-GUID headers.

<131> Section 2.2.2.18: Outlook 2010 sets the PidTagAttachPayloadClass and PidTagAttachPayloadProviderGuidString properties, but sets them on the attachment instead of the message.

<132> Section 2.2.2.18: Outlook 2010 does not read the X_Payload-Class and X-Payload-Provider-GUID MIME headers when they appear on a MIME entity that is analyzed as a message or message body, rather than as an attachment.

<133> Section 2.2.2.19: Office Outlook 2003 and Office Outlook 2007 neither read nor write the PidTagPurportedSenderDomain **property**.

<134> Section 2.2.2.20: Office Outlook 2003 and Office Outlook 2007 neither read nor write the PidTagSenderIdStatus **property**.

<135> Section 2.2.2.21: Office Outlook 2003 and Office Outlook 2007 neither read nor write the PidTagContentFilterSpamConfidenceLevel **property**.

<136> Section 2.2.2.22: Office Outlook 2007 and Outlook 2010 neither read nor write the X-Microsoft-Classified header.

<137> Section 2.2.2.22: Office Outlook 2003 and Office Outlook 2007 do not read or set any of the classification headers.

<138> Section 2.2.2.22: Exchange 2003 does not read or set any of the classification headers.

<139> Section 2.2.2.23: Office Outlook 2003 does not support unified messaging.

<140> Section 2.2.2.23: Office Outlook 2003 and Office Outlook 2007 do not read or set any of the unified messaging headers.

<141> Section 2.2.2.23: Exchange 2003 does not read or set any of the unified messaging headers.

<142> Section 2.2.2.24: Office Outlook 2003 and Office Outlook 2007 do not copy the value of the Content-ID header to PidTagBodyContentId and do not copy the value of PidTagBodyContentId to the Content-ID header.

<143> Section 2.2.2.24: Exchange 2003 and Exchange 2007 do not copy the value of the Content-ID header to PidTagBodyContentId and do not copy the value of PidTagBodyContentId to the Content-ID header. Exchange 2010 copies the value of the Content-ID header to PidTagBodyContentId.

<144> Section 2.2.2.25: Office Outlook 2003 and Office Outlook 2007 do not copy the value of a Content-Base header to the PidNameContentBase property.

<145> Section 2.2.2.26: Exchange 2003 does not support the PidTagBodyContentLocation property.

<146> Section 2.2.2.26: Exchange 2007 does not copy the value of PidTagBodyContentLocation to the Content-Location header.

<147> Section 2.2.2.26: Office Outlook 2003 and Office Outlook 2007 do not copy the value of a Content-Location header to the PidTagBodyContentLocation property.

<148> Section 2.2.2.27: Outlook 2010 does not read the XRef header.

<149> Section 2.2.2.28: Office Outlook 2003 and Office Outlook 2007 also set PidTagTransportMessageHeaders when **downloading** messages via POP3 or IMAP.

<150> Section 2.2.2.29: Both Exchange 2003 and Exchange 2007 cannot create the named property corresponding to a generic header if (1) the mailbox database named property quota has been exceeded; or (2) measures adopted to prevent exhausting the named property quota interfere with creating the named property.

<151> Section 2.2.2.29: Office Outlook 2003 and Office Outlook 2007 do not create named properties for headers.

<152> Section 2.2.2.29: Exchange 2003 does not support setting ad-hoc headers as named properties.

<153> Section 2.2.4.1.1: Exchange 2003 replaces all of the illegal characters with the underscore "_" (U+005F), except for the backslash "\" (U+005C), which is removed.

<154> Section 2.2.4.1.1: Exchange 2007 does not strip control characters U+0001 through U+0004 from the attachment filename.

<155> Section 2.2.4.1.1: Exchange 2003 does not perform the following steps when it creates PidTagAttachLongFilename and PidTagDisplayName.

<156> Section 2.2.4.1.1: Exchange 2007 removes leading spaces from the start of the base. It removes trailing spaces from the end of the extension. It does not remove "." (U+002E) characters from the base.

<157> Section 2.2.4.1.1: Exchange 2007 does not generate an extension for PidTagAttachLongFilename when only the name portion is in MIME. Exchange 2003 copies a sanitized version of the filename to PidTagAttachLongFilename; it generates a base and extension only when there is no MIME filename.

<158> Section 2.2.4.1.1: Exchange 2003 does not replace an empty display name with the base part of the file name. The display name contains only the file extension.

<159> Section 2.2.4.1.1: Exchange 2003 copies PidTagAttachLongFilename from the MIME header value after removing invalid filename characters. Only step 1 of the included list of seven steps is completed before the MIME header value is copied.

<160> Section 2.2.4.1.1: When the first character of the MIME header value is "." (U+002E), Exchange 2003 considers the name part to be empty, but not the extension part.

<161> Section 2.2.4.1.1: Exchange 2003 also replaces the question mark "?" (U+003F) and the asterisk "*" (U+002A) with the underscore "_" (U+005F).

<162> Section 2.2.4.1.1: Exchange 2003 removes plus "+" (U+002B), equal "=" (U+003D), left square bracket "[" (U+005B), right square bracket "]" (U+005D), and semicolon ";" (U+003B).

<163> Section 2.2.4.1.1: Exchange 2003 does not delete the apostrophe "'" (U+0027), and replaces the question mark "?" (U+003F) and asterisk "*" (U+002A) with the underscore "_" (U+005F)

<164> Section 2.2.4.1.1: Exchange 2003 sets PidTagAttachFilename to "NONAME" when PidTagAttachLongFilename is empty.

<165> Section 2.2.4.1.1: Exchange 2003 deletes the extension portion of the filename when copied to PidTagAttachFilename when the original extension is greater than three characters in length.

<166> Section 2.2.4.1.1: Exchange 2003 truncates the name part of PidTagAttachLongFilename to 8 characters and does not add "~1".

<167> Section 2.2.4.1.1: Exchange 2003 does not generate the extension part of PidTagAttachFilename.

<168> Section 2.2.4.1.2: Exchange 2003 only replaces the "application/ms-TNEF" value with "application/octet-stream" when it is content-type for the root body part.

<169> Section 2.2.4.1.3: Office Outlook 2003 and Office Outlook 2007 do not use the parameters of the Content-Disposition header to set creation or modification dates.

<170> Section 2.2.4.1.3: Exchange 2007 uses current system time to set PidTagCreationTime and PidTagLastModificationTime when the Content-Disposition header values are missing or invalid.

<171> Section 2.2.4.1.3: Exchange 2003 uses current system time to set PidTagLastModificationTime.

<172> Section 2.2.4.1.4: Exchange 2003 does not map the Content-Base header to PidTagAttachContentBase on attachments.

<173> Section 2.2.4.1.4: Exchange 2007 MIME reader sets the value of PidTagAttachFlags to "0x00000001" for inline attachments.

<174> Section 2.2.4.1.4: The MIME reader in Exchange 2007 does not verify that inline attachment candidates are in fact referenced from the message body before marking them as inline.

<175> Section 2.2.4.2.1: The PidNameAttachmentMacContentType property is not written by Exchange 2003.

<176> Section 2.2.4.2.1: The PidNameAttachmentMacInfo property is not written by Exchange 2003.

<177> Section 2.2.4.2.1: Exchange 2003 does not reject the message as not MIME compliant when parsing of the header part fails.

<178> Section 2.2.4.2.1: Exchange 2007 gets the attachment file name from AppleSingle data, instead of preferring a file name found in MIME headers.

<179> Section 2.2.4.2.2: Exchange 2003 does not reject the message when the MIME reader fails to parse the MIME body.

<180> Section 2.2.4.2.2: Exchange 2007 gets the attachment file name from AppleSingle or MacBinary data, instead of preferring a file name found in MIME headers.

<181> Section 2.2.4.2.2: Exchange 2003 does not map the MacBinary version number (MacBinary header offset 122) or the minimum MacBinary version number (MacBinary header offset 123) into the PidTagAttachDataBinary property.

<182> Section 2.2.4.2.2: Exchange 2007 does not support a MacBinary structure with additional header data or comment part present.

<183> Section 2.2.4.2.2: Exchange 2003 uses a maximum field length of 62 bytes for file name, not 63 bytes.

<184> Section 2.2.4.2.2: Exchange 2003 does not set the signature.

<185> Section 2.2.4.2.2: Exchange 2003 uses bytes 75:76 to map the horizontal location of the icon.>

<186> Section 2.2.4.2.2: Exchange 2003 uses bytes 77:78 to map the vertical location of the icon.

<187> Section 2.2.4.2.2: Exchange 2003 and Exchange 2007 conversion **handle** only EntryID values of "1", "2", "3", "8", and "9". Other EntryID values are ignored.

<188> Section 2.2.4.2.3: Exchange 2007 does not ignore the Content-Transfer-Encoding header when dealing with BinHex60 attachments. If the Content-Transfer-Encoding header is present, then Exchange 2007 uses the encoding method specified in the header to process the data.

<189> Section 2.2.4.2.3: Exchange 2003 does not decode the attachment data and sets PidTagAttachDataBinary to the encoded binhex40 attachment data. Exchange 2003 does not set the value of PidNameAttachmentMacInfo.

<190> Section 2.2.4.2.3: Exchange 2003 does not parse the binhex40-encoded data, and does not detect or reject data that is not MIME compliant.

<191> Section 2.2.4.2.3: Exchange 2007 gets the attachment file name from BinHex data, instead of preferring a file name found in MIME headers.

<192> Section 2.2.4.2.3: Exchange 2003 generates the PidTagDisplayName and PidTagAttachLongFilename properties from BinHex data, instead of preferring a file name found in MIME headers. The PidTagAttachFilename property is generated from the MIME headers.

<193> Section 2.2.4.3: Exchange 2003 uses the Subject header in preference to the Content-Description header value when generating the PidTagDisplayName property.

<194> Section 2.2.4.3: Exchange 2003 does not write the file name value of an embedded attachment to the PidTagAttachLongFilename property.

<195> Section 2.2.4.3: Exchange 2003 does not save the resulting extension value in the PidTagAttachExtension property.

<196> Section 2.2.4.3: Exchange 2003 does not generate the PidTagAttachFilename for embedded message attachments.

<197> Section 2.2.4.3: Office Outlook 2003 and Office Outlook 2007 do not save the X-MS-Exchange-Organization-Original-Sender header value.

<198> Section 2.2.4.3: Exchange 2003 does not save the X-MS-Exchange-Organization-Original-Sender header value.

<199> Section 2.2.4.3: Office Outlook 2003 and Office Outlook 2007 exclude unknown MIME headers that start with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-" from analysis.

<200> Section 2.2.4.3: Exchange 2003 excludes unknown MIME headers that start with "X-MS-Exchange-Organization-" or "X-MS-Exchange-Forest-" from analysis.

<201> Section 2.2.4.4.1: vCard Version 2.1 is used by Office Outlook 2003, Office Outlook 2007, and Outlook 2010. vCard Version 2.1 is supported on incoming messages by Exchange 2010, but not by Exchange 2003 and Exchange 2007.

<202> Section 2.2.5: Office Outlook 2003 and Office Outlook 2007 analyze attachment MIME parts with the Content-Type set to "message/external-body" the same as they do ordinary file attachments. No special analysis is performed.

<203> Section 2.2.5: Exchange 2007 does not append the "URL" extension when the name is empty.

<204> Section 2.2.5: Exchange 2003 sanitizes the external body attachment file name by replacing the following characters with the underscore "_" (U+005F): double quote """ (U+0022), forward slash "/" (U+002F), colon ":" (U+003A), left angle bracket "<" (U+003C), right angle bracket ">" (U+003E), pipe "|" (U+007C), and backslash "\" (U+005C).

<205> Section 2.2.5: Exchange 2003 sets PidTagAttachFilename to "NONAME.URL" when the base part of the external body attachment file name is empty. PidTagAttachLongFilename contains no base part name and the extension is set to "URL".

<206> Section 2.2.6: Outlook 2010 does not copy data into the PidTagMimeSkeleton property.

<207> Section 2.3.2: Exchange 2003, Exchange 2007, Exchange 2007 SP1, and Exchange 2007 SP2 do not reject a message containing the Content-Type header of message/partial.

<208> Section 2.4: Exchange 2007 does not have this functionality. Exchange 2003 does not have this functionality, but does save the entire original MIME message.

<209> Section 2.4: Outlook 2010 does not create the PidTagMimeSkeleton property ([MS-OXPROPS] section 2.902).

# 6 Change Tracking

This section identifies changes that were made to the [MS-OXCMAIL] protocol document between the August 2010 and November 2010 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- Changes made for template compliance.
- Removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type "Editorially updated."

Some important terms used in revision type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 1.1 Glossary | 57983<br>Added "Contact object" to the list of terms defined in [MS-OXGLOS]. Added a definition for the term "contact attachment". | N | Content updated. |
| 1.1 Glossary | 56380<br>Added "one-off address" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56382<br>Added term "delivery status notification" to local glossary. | N | Content updated. |
| 1.1 Glossary | 56386<br>Added "domain" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56580<br>Added "Calendar object" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56579<br>Added "reminder" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56384<br>Added "recipient table" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 1.1 Glossary | 56397<br>Added "mailbox" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56398<br>Added "attachments table" to the list of terms defined by [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56569<br>Added "address list" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56396<br>Added "property type" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56572<br>Added "GUID" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56393<br>Added "property name" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56383<br>Added "Post Office Protocol – Version 3 (POP3)" to the list of terms defined by [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56404<br>Added "contact" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56403<br>Added "meeting" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56561<br>Added "metafile" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56408<br>Added "MIME content-type" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56574<br>Added "flags" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56543<br>Added "S/MIME" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.1 Glossary | 56571<br>Added "named property" to list of terms | N | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| | defined in [MS-OXGLOS]. | | |
| 1.1 Glossary | 57401 Added "Messaging Application Programming Interface (MAPI)" to the list of terms defined in [MS-OXGLOS]. | N | Content updated. |
| 1.2.1 Normative References | 57761 Added reference [UNICODE]. | N | New content added. |
| 1.2.1 Normative References | 57661 Added reference to [RFC1738]. | N | New content added. |
| 2 Structures | 56965 Changed "two symmetrical parts" to "two parts". | N | Content updated. |
| 2.1 MIME Generation | 56380 Changed "one-off recipient" to "message sent using a one-off address". | N | Content updated. |
| 2.1.1.1 Recipients | 58404 Altered steps 3, 4, 5 and 6 to eliminate a logical contradiction in the instructions. | Y | Content updated. |
| 2.1.1.1 Recipients | 57248 Changed "non-delivery receipt" to "non-delivery report" and linked to term definition in [MS-OXGLOS]. | N | Content updated. |
| 2.1.1.1 Recipients | 57254 Linked to section of [MS-OXOMSG] that specifies the properties for non-delivery reports. | N | Content updated. |
| 2.1.1.1 Recipients | 56543 Added section links for [MS-OXCDATA] and [MS-OXOABK] references. | N | Content updated. |
| 2.1.1.1.1 To and Cc Recipients | 56382 Linked "delivery status notification" to the local glossary. | N | Content updated. |
| 2.1.1.1.1 To and Cc Recipients | 57256 Clarified that MIME writers use the PidTagReceivedBy group of the attached message if the PidTagReceivedRepresenting group is not defined. | N | Content updated. |
| 2.1.1.2 Reply-to | 57280 Added link to one-off EntryID specification in [MS-OXCDATA]. | N | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 2.1.1.4 Sender | 56585 Clarified which properties belong to the PidTagSender group. | N | Content updated. |
| 2.1.1.5 Return Receipt | 58046 Updated content to reflect current normative behavior. Created a product behavior note specifying that Exchange 2007 uses the PidTagSentRepresenting group instead. | Y | Content updated. |
| 2.1.1.6 Read Receipt | 56586 Specified which properties belong to the PidTagReadReceipt and PidTagSentRepresenting groups. | N | Content updated. |
| 2.1.1.8 IMCEA Encapsulation | 56387 Clarified which domains should be used with IMCEA encapsulation. | N | Content updated. |
| 2.1.2 Envelope Elements | 56388 Specified the conditions under which MIME headers are encoded. | N | Content updated. |
| 2.1.2.1 Message Class | 57258 Updated IPM.Appointment entry to reflect that values both equal to and containing this string meet the specified criteria. | N | Content updated. |
| 2.1.2.2 Content Class | 56749 Clarified the circumstance under which MIME writers look up the value of the PidNameContentClass property. | N | Content updated. |
| 2.1.2.2 Content Class | 56543 Broke one large table into two separate tables with distinctly labeled header rows. | N | Content updated. |
| 2.1.2.4 Arbitrary MIME Headers | 56388 Specified the conditions under which header values are encoded. | N | Content updated. |
| 2.1.2.7 Sent Time | 57113 Added section reference for [RFC2822]. | N | Content updated. |
| 2.1.2.8 Subject | 56395 Removed sentence about typical truncation size. | N | Content updated. |
| 2.1.2.19 Has Attach | 58037 Changed "X-MS-HasAttach" to "X-MS-Has-Attach." | Y | Content updated. |
| 2.1.2.19 Has Attach | 56398 Changed "attachment table" to | N | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| | "attachments table" | | |
| 2.1.2.20 Auto Response Suppress | 56581 Changed Unicode value for comma from U+0032 to U+002C. | N | Content updated. |
| 2.1.2.20 Auto Response Suppress | 57800 Changed value from "0x000000020" to "0x00000020". | N | Content updated. |
| 2.1.2.27 Received Headers | 49714 Clarified the circumstances under which MIME writers should copy all Received header fields from PidTagTransportMessageHeaders to the generated MIME header. | N | Content updated. |
| 2.1.2.27 Received Headers | 57257 Moved content pertaining to MIME analysis to PidTagTransportMessageHeaders section. | N | Content updated. |
| 2.1.2.30 XRef | 55948 Clarified that we are discussing MIME writer behavior in this section. | Y | Content updated. |
| 2.1.3.1 Client Actions | 58078 Added product behavior note detailing how Exchange 2007 handles conflicting charset values on different body formats. | N | Content updated. |
| 2.1.3.1 Client Actions | 56583 Changed "binary" and "string" to "PtypBinary" and "PtypString", respectively. | N | Content updated. |
| 2.1.3.2 Message Body in TNEF | 56401 Specified under what conditions a TNEF plain text body should be converted from the best body. | N | Content updated. |
| 2.1.3.8 Calendar Items and Meeting Messages | 57258 Clarified that only values of PidTagMessageClass that start with "IPM.Schedule.Meeting." are meeting messages. | Y | Content updated. |
| 2.1.4 Attachments | 56405 Moved example code from this topic to Structure Examples section. | Y | Content updated. |
| 2.1.4.1 Inline Attachments | 56406 Listed out specific properties on attachments that MIME writers should ignore when the body of the message is plain text. | Y | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 2.1.4.1.1 Inline Attachments in RTF Messages | 56584 Changed "0x0000006" to "0x00000006." | N | Content updated. |
| 2.1.4.2.2 Content-Type, Content-Description, Content-Disposition | 56409 Clarified the conditions under which Content-Description should be encoded. | N | Content updated. |
| 2.1.4.3 MacBinary Attached Files | 57585 Removed a paragraph about the format of multipart/appledouble. | N | Content removed. |
| 2.1.5 Generating Pure MIME Messages | 56573 Clarified that this section applies only if PidTagMimeSkeleton is defined. | Y | Content updated. |
| 2.1.5.1 Generation Process | 56573 Clarified how attachments and body parts are mapped from the MIME skeleton to message body parts in the object model. | Y | Content updated. |
| 2.2.2.6 Subject | 56395 Removed sentence about typical truncation size. | N | Content updated. |
| 2.2.2.15 Content Class | 56748 Specified where to end the extracted substring for a Content-Class prefixed with "InfoPath.". | N | Content updated. |
| 2.2.2.18 Payload Properties | 57984 Changed recommendation for MIME readers on copying X-Payload-Class and X-Payload-Provider-GUID classes from "SHOULD ignore" to "SHOULD copy". Added product behavior note for Outlook 2010. | Y | Content updated. |
| 2.2.2.20 Sender Id Status | 58887 Moved mapping table for header values from [MS-OXCSPAM] into this specification. | Y | Content updated. |
| 2.2.2.27 XRef | 55948 Removed MIME writer behavior from this section. | Y | Content updated. |
| 2.2.2.28 PidTagTransportMessageHeaders | 57257 Removed references to "client". Added information from the "MIME Generation" section. | N | Content updated. |
| 2.2.3.2.1 Selecting the Primary Message Text MIME Element | 58156 Changed "can" in "MIME readers can use the value of a Content-Type meta tag" to "MAY". | Y | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 2.2.3.2.1 Selecting the Primary Message Text MIME Element | 56568 Clarified that the specified content types are listed in descending order of preference. | N | Content updated. |
| 2.2.4 Attachments | 57636 Added link to section that specifies selecting the MIME entity that represents the message body. | Y | Content updated. |
| 2.2.4.1.1 File name | 57761 Added reference to [UNICODE]. | N | Content updated. |
| 2.2.4.1.1 File name | 57747 Clarified that name should have "~1" entered only when name is truncated, not when name or extension are truncated. | N | Content updated. |
| 2.2.4.2.1 Multipart/Appledouble | 58111 Provided link to clarify statement about MIME part analysis overwriting previous MIME property value settings. | N | Content updated. |
| 2.2.4.2.2 Application/Applefile | 56391 Added missing field length values. | N | Content updated. |
| 2.2.4.3 Attached Messages | 57806 Clarified what values to use when the base filename is turned into an empty string. | N | Content updated. |
| 2.2.4.3 Attached Messages | 58030 Changed value of mfRead flag after MIME analysis from 0x1 to 0x0. | Y | Content updated. |
| 2.2.5 External Body Attachments | 57661 Moved reference to [RFC1738] out of code and into the document text. | N | Content updated. |
| 2.3.1 Analysis of Non-MIME Content | 56576 Clarified how MIME readers can assume the presence of a MIME-Version field. | N | Content updated. |
| 2.3.2 Message/Partial | 57910 Added Exchange 2007, Exchange 2007 SP1, and Exchange 2007 SP2 to the list of products that do not support blocking the message/partial content type. | Y | Content updated. |
| 2.4.3 MIME Conversion | 56390 Changed "DSN/MDN" to "delivery status notification", and linked term to local glossary. | N | Content updated. |
| 2.4.3 MIME Conversion | 56543 Changed "SMIME" to "S/MIME". | N | Content updated. |

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change Type |
|---|---|---|---|
| 3.1 MIME Examples | 56405 Added section. | N | Content updated. |
| 3.1.1 MIME Message Containing Inline and Non-Inline Attachments | 56405 Added section. | N | Content updated. |
| 3.1.2 MIME Message Containing Only Inline Attachments | 56405 Added section. | N | Content updated. |
| 3.1.3 MIME Message Containing Only Non-Inline Attachments | 56405 Added section. | N | Content updated. |
| 4.1 Unsolicited Commercial E-mail (Spam) | 56543 Revised description of physical bulk mail and description of handling external addresses. | N | Content updated. |

# 7 Index


**C**

Change tracking 99
Common data types and fields 15

**D**

Data types and fields - common 15
Details
  common data types and fields 15

**E**

Example 80

**F**

Fields - vendor-extensible 14

**G**

Glossary 7

**I**

Informative references 12
Introduction 7

**L**

Localization 14

**N**

Normative references 9

**O**

Overview (synopsis) 12

**P**

Product behavior 87

**R**

References
  informative 12
  normative 9
Relationship to protocols and other structures
  (section 1.4 13, section 1.5 13)

**S**

Structures
  overview 15

**T**

Tracking changes 99

**V**

Vendor-extensible fields 14
Versioning and Localization 14