[MS-OXOMSG]: E-mail Object Protocol Specification

Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights**. This protocol documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the protocols, and may distribute portions of it in your implementations of the protocols or your documentation as necessary to properly document the implementation. This permission also applies to any documents that are referenced in the protocol documentation.

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

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

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

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

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

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

Revision Summary			
Author	Date	Version	Comments
Microsoft Corporation	April 4, 2008	0.1	Initial Availability.
Microsoft Corporation	April 25, 2008	0.2	Revised and updated property names and other technical content.

Table of Contents

1 Introduction

An e-mail client provides the user interface for composing, reading and sending messages, and for accessing and modifying the message items contained in message stores. An e-mail object represents a single message in a folder of the message store used to send or receive e-mail.

The E-mail Object Protocol specifies:

- The properties of a message object in the message store mailbox
- The transport features specific to an e-mail message object

1.1 Glossary

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

ANSI character set

ASCII

best body

blind carbon copy recipient

carbon-copy recipient

Drafts folder

FAI

folder object

from properties

GUID

Inbox folder

IPM

Message object

MIME

Outbox folder

primary recipient

property (3)

recipient object

remote procedure call (RPC)

Rich Text Format (RTF)

ROP request buffer

ROP response buffer

search folder

sender properties

Sent Mail folder

store

Transport Neutral Encapsulation Format (TNEF)

Unicode

The following terms are specific to this document:

conversation thread: A series of messages and their responses (usually related by subject).

e-mail object: A **message object** that represents an e-mail message in a messaging store and that adheres to the **property** specifications in this document. An **e-mail object** models the electronic equivalent of mail.

mail spooler: A program or function that receives requests to send mail to and deliver mail for a user. It determines which mail transport handles the sending or receipt of mail.

messaging transport: A networking protocol that facilitates the transfer of messages between a messaging client and a messaging server.

recipient properties: A group of properties that identify an intended recipient or recipients of a message.

resend message: A message that is submitted for message delivery after it has failed to be sent to all or some of its recipients.

spooler queue: A series of outgoing messages that are ready for delivery to recipients.

UUENCODED attachments: See [IEEE1003.1]

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

1.2 References

1.2.1 Normative References

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

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

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

[MS- OXCFXICS] Microsoft Corporation, "Bulk Data Transfer Protocol Specification", April 2008.

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

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

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

[MS-OXCSTOR] Microsoft Corporation, "Store Object Protocol Specification", April 2008.

[MS- OXCTABL] Microsoft Corporation, "Table Object Protocol Specification", April 2008

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

[MS-OXMSG] Microsoft Corporation, ".MSG File Format Specification", April 2008.

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

[MS-OXODLGT] Microsoft Corporation, " Delegate Access Configuration Protocol Specification", April 2008.

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

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

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2921. 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

1.2.2 Informative References

None.

1.3 Protocol Overview (Synopsis)

The E-mail Object Protocol specifies the representation of an e-mail message in a messaging store. The E-mail Object Protocol extends the Message and Attachment Object Protocol in that it defines new properties and adds restrictions to the properties that are specified in [MS-OXCMSG].

An **e-mail object** represents a single e-mail message. The properties that are specific to an e-mail object facilitate retaining information about the e-mail message's sender, recipients, subject, message content, and all the options associated with this e-mail set by the sender or recipient. An e-mail object is stored in a **folder object**. The E-mail Object Protocol also specifies how an e-mail object is used to represent a special type of message: Report Message, which is generated to report the status of a sent message either at the sender's request or at the request of the system administrator.

1.3.1 E-mail Objects

1.3.1.1 Creating, Opening, and Saving E-mail Objects

E-mail objects adhere to the specifications in [MS-OXCMSG].

1.3.1.2 Sending Messages

A client submits a request to a server to send an e-mail message to another messaging user. The server can defer or reject the request based on the properties and permissions associated with the e-mail object.

While the message is queued in the server, the client can abort the send operation.

1.3.1.3 Replying and Forwarding Messages

Replying to a message or forwarding a message is identical to sending a message except that both actions have an expanded set of properties that are specified in section 2.2.1.

1.3.2 Report Messages

Report messages are an extension of the e-mail object. Report messages present status information about a sent message to its sender. There are two general types of reports:

- **Read status reports**: Read receipt reporting occurs when the sent e-mail is read/opened by the recipient and Nonread receipt reporting occurs when the sent e-mail is not read before it is deleted or expired.
- **Delivery status reports**: Delivery receipt reporting occurs when the sent e-mail is delivered to the recipient and Nondelivery receipt reporting occurs when the sent e-mail cannot be delivered.

1.3.2.1 Read Receipt

A read receipt report indicates that a sent e-mail message was read or opened by a recipient.

Read receipts are not generated automatically. Senders that want to receive read receipts explicitly request them.

1.3.2.2 Nonread Receipt

A nonread receipt is generated during e-mail message deletion operations as defined in [MS-OXCFOLD], at the expiration of a time limit, or according to client specific criteria. A nonread receipt is sent to the e-mail's sender or a designated recipient by the e-mail sender's request.

1.3.2.3 Delivery Receipt

A delivery receipt is generated and sent by the messaging system to the e-mail's sender or designated recipient when an e-mail has reached its intended recipient.

1.3.2.4 Non-Delivery Receipt

The nondelivery receipt is generated and sent to the e-mail's sender by the messaging system when an e-mail could not reach an intended recipient. Nondelivery receipts are sent automatically unless a request is made to suppress them.

1.3.2.5 Voting and Tracking

Voting and Tracking are an extension of the e-mail object. When composing a survey-type e-mail message, a client can add voting options to the e-mail message by setting voting verb properties as specified in section 2.2.1.60 on an outgoing message and send it to recipients. A recipient's client can respond to the voting survey by setting response properties on a reply message. The sender's client processes the reply message and maintains the response tracking information in the original message's recipient tracking status properties, as specified in section 2.2.1.61.

1.3.3 Controlling sending and delivery of mail

If a client is connected to several e-mail servers at once (not necessarily using the same protocol), it can choose to control sending of mail by manipulating the spooler queue of the message store. If a client delivers mail into a folder on the server (such as delivering POP3 messages), it can inform the server of the new mail through ROP requests.

1.4 Relationship to Other Protocols

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

1.5 Prerequisites/Preconditions

The E-mail Object Protocol specification is an extension of [MS-OXCMSG], and no further prerequisites or preconditions exist.

1.6 Applicability Statement

The E-mail Object Protocol is used to model the exchange of interpersonal mail and messages.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The E-mail Object Protocol uses the protocols specified in [MS-OXCPRPT] and [MS-OXCMSG] as its primary transport mechanism.

The **ROP Request Buffers** and **ROP Response Buffers** specified by this protocol are respectively sent to and received from the server using the underlying **remote procedure call (RPC)** transport as specified in [MS-OXCROPS].

2.2 Message Syntax

An **e-mail object** can be created and modified by clients and servers. Except where noted below, this section defines constraints to which both clients and servers MUST adhere when operating on e-mail objects.

Clients operate on e-mail objects by using the Message and Attachment Object Protocol [MS-OXCMSG]. How a server operates on e-mail objects is implemention-dependent, but the results of any such operations MUST be exposed to clients in a manner that is consistent with the E-mail Object Protocol.

Unless otherwise specified below, e-mail objects adhere to all property constraints specified in [MS-OXPROPS] and all property constraints specified in [MS-OXCMSG]. An e-mail object can also contain other properties as described in [MS-OXPROPS], but these properties have no impact on this protocol.

When a property is referred to as "read-only for the client" the server MUST return an error and ignore any request to change the value of that property.

2.2.1 E-mail Message Object Properties

The following properties are specific to e-mail objects.

2.2.1.1 PidTagBlockStatus

Type: PtypInteger32

Indicates the user's preference for viewing external content (such as links to images on an HTTP server) in the message body. A client MAY ignore this value and always allow or block external content based on other factors (such as whether the sender is on a safe list). If this property is used, then the default action is to block the external content. However, if the value of this property falls within a certain range, then viewing external content is allowed. The allowed value is computed from PidTagMessageDeliveryTime: since the sender of a message does not have knowledge of this value, the sender cannot reliably set PidTagBlockStatus to the allowed values.

To compute the allowed values, convert the value of PidTagMessageDeliveryTime to a PtypDouble, *floatdate,* where the date is represented as the number of days from midnight, December 30, 1899. Apply the following formula:

result = ((*floatdate* - *floor*(*floatdate*)) * 100000000) + 3;

where *floor*(x) returns the largest integer ≤ x.

Convert the PtypDouble value *result* to a 32-bit integer *computedvalue*.

Clients SHOULD set PidTagBlockStatus to *computedvalue* to allow external content. However, when determining whether to accept external content, clients SHOULD allow external content if the absolute value of the difference between *computedvalue* and the value of PidTagBlockStatus is 1 or less.

2.2.1.2 **PidTagConversationIndex**

Type: PtypBinary

Indicates the relative position of this message within a conversation thread. It MUST be set according to the description in the following table:

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>1
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>2
0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>3
0</th><th>1</th></tr>
<tr><td colspan="32">Conversation Index Header (22 bytes)</td></tr>
<tr><td colspan="32">Conversation Index Header (continued)</td></tr>
<tr><td colspan="32">Conversation Index Header (continued)</td></tr>
<tr><td colspan="32">Conversation Index Header (continued)</td></tr>
<tr><td colspan="32">Conversation Index Header (continued)</td></tr>
<tr><td colspan="24">Conversation Index Header (continued)</td><td colspan="8">Response Level 1(5 bytes)</td></tr>
<tr><td colspan="28">Response Level 1 (Continued)</td><td colspan="4">…</td></tr>
</table>

…
Respsonse Level N (5 bytes)
Respsonse Level N (continued)

Conversation Index Header (22 bytes):

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	0	0	0	0	0	1	*Current FILETIME high part – low 24 bits*																							
Current FILETIME low part – high 16 bits																*Guid -2 bytes*														
Guid (continued)-4 bytes																															
Guid (continued)-4 bytes																															
Guid (continued)-4 bytes																															
Guid(continued)-2 bytes																															

Current FILETIME: The current time in UTC expressed as a PtypTime is obtained, where only the 24 low bits of the high part and the 16 high bits of the low part of the FILETIME are included in *Current FILETIME high part* and *Current FILETIME low part*, as shown in the following table:

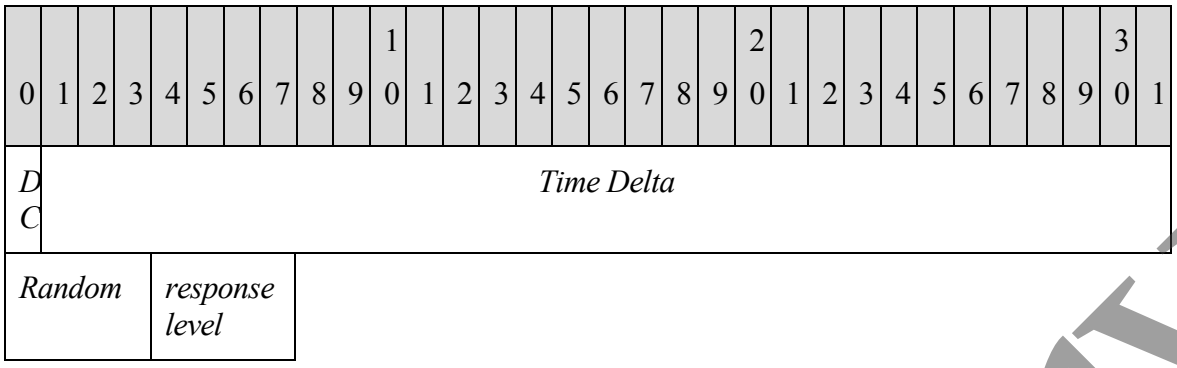
Table 1 FILETIME bits included in Conversation Index Header

8 most significant bits	40 bits	16 least significant bits
Excluded	Included	Excluded

The data is stored in big-endian format – the 5 bytes of the time are written from most significant byte to least significant byte.

Guid (16 Bytes, PtypGuid): Generated for each new conversation thread. The Data1, Data2, and Data3 fields are stored in big-endian format in the packet.

Response Levels (5 bytes each):



DC (Delta code) and **Time Delta** are calculated based on the difference between the current time and the time stored in the Conversation Index header:

- If the difference is less than 1.7 years (high order part of the delta file time bitwise AND with 0x00FE0000 resulting in 0), the delta code is 0 and the time delta is the least significant 31 bits of the difference remaining once the 18 least significant bits are excluded:

15 most significant bits	31 bits	18 least significant bits
Excluded	Included	Excluded

- If the difference is greater than or equal to 1.7 years (high order part of the delta file time bitwise AND with 0x00FE0000 resulting non-zero), the delta code is 1 and the time delta is the least significant 31 bits of the difference remaining once the 23 least significant bits are excluded.

10 most significant bits	31 bits	23 least significant bits
Excluded	Included	Excluded

For both cases, Time Delta is stored in big-endian format.

Random: 4 bits generated using an implementation-specific algorithm.

Response level: This field SHOULD always be set to all 0's.

2.2.1.3 **PidTagConversationTopic**

Type: PtypString

Contains an unchanging copy of the original subject; it MUST be set to the same value as PidTagNormalizedSubject, as specified in [MS-OXCMSG] on an e-mail object when it is submitted.

2.2.1.4 **PidTagDeferredDeliveryTime**

Type: PtypTime

Contains the date and time, in UTC, at which the sender prefers the message to be delivered. This property MAY be absent; if so, the message is delivered as soon as possible. If present, it MUST have the same value as PidTagDeferredSendTime specified in section 2.2.3.5<1>.

2.2.1.5 PidTagDisplayBcc

Type: PtypString

MUST be set to a list of the display names of **blind carbon copy** recipients separated by semicolons if an e-mail has blind carbon copy recipients. Otherwise, this property MUST contain an empty string as specified in [MS-OXCMSG]. This property is read-only for the client.

2.2.1.6 PidTagDisplayCc

Type: PtypString

MUST be set to a list of the display names of **carbon copy recipients** separated by semicolons if an e-mail has carbon copy recipients. Otherwise, this property MUST contain an empty string as specified in [MS-OXCMSG]. This property is read-only for the client.

2.2.1.7 PidTagDisplayTo

Type: PtypString

MUST be set to a list of the display names of the **primary recipients** separated by semicolons if an e-mail has primary recipients. Otherwise, this property MUST contain an empty string as specified in [MS-OXCMSG]. This property is read-only for the client.

2.2.1.8 PidTagIconIndex

Type: PtypInteger32

Specifies which icon is to be used by a user interface when displaying a group of e-mail objects. This property, if it exists, is a hint to the client: it MAY<2> ignore the value of this property and use another method of determining what icon to show to the user (such as using the value of PidTagMessageClass or PidTagMessageFlags). Table 1 shows examples of PidTagIconIndex values.

Table 2 Examples of PidTagIconIndex values

Mail Item State	Mail Item Icon Index
New mail	0xFFFFFFFF
Read mail	0x00000100
Unread mail	0x00000101
Submitted mail	0x00000102

Unsent mail	0x00000103
Receipt mail	0x00000104
Replied mail	0x00000105
Forwarded mail	0x00000106
Remote mail	0x00000107
Delivery Receipt	0x00000108
Read Receipt	0x00000109
Nondelivery Receipt	0x0000010A
Nonread Receipt	0x0000010B
Recall_S mails	0x0000010C
Recall_F mail	0x0000010D
Tracking mail	0x0000010E
Out of Office mail	0x0000011B
Recall mail	0x0000011C
Tracked mail	0x00000130

2.2.1.9 **PidTagInternetMailOverrideFormat**

Type: PtypInteger32

Indicates the encoding method and HTML inclusion for attachments and SHOULD be set on outgoing mail. This property is broken up into sub-portions as shown in the following table. Note that "X" indicates that the bit is not to be set, and if set, the bit is to be ignored; the format of the table is little-endian:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
x	x	x	x	x	*Format 1*			x	x	x	x	x	x	x	x	x	x	x	*E18*		*M4*	*P2*	x	x	x	x	x	x	x	x	x

***Format1* (3 bits)**: MUST be set to one of the following values:

Value	Meaning
0x0	Default – The mail system chooses the default encoding scheme, based on other fields in this property value.
0x1	The message is sent as a **MIME** with text/plain and text/html body parts.
0x2	The message is sent as plain text with UUENCODED attachments.
0x4	The message is sent as a **MIME** with text/plain and text/html body parts (for example, treat as 0x1).

***P2* (1 bit)**: MUST be ignored if *Format1* == 0; otherwise, indicates the preference, as follows:

Value	Meaning
0	Ignore *M4*.
1	Use *M4* to determine encoding.

***M4* (1 bit)**: MUST be ignored if *Format1* == 0 or *P2* == 0; otherwise, indicates the encoding, as follows:

Value	Meaning
0	Use UUENCODE, and ignore *E18*.
1	Use MIME encoding, and use *E18* to determine body inclusions.

***E18* (2 bits)**: MUST be ignored if *Format1* == 0 or *P2* == 0 or *M4* == 0. Otherwise, MUST be one of the following values to indicate the HTML inclusion:

Value	Meaning
0x0	Text/plain only.
0x1	Text/plain and text/html (for example, treat as identical to 0x2).
0x2	Text/plain and text/html.

2.2.1.10 **PidTagInternetMessageId**

Type: PtypString

Corresponds to the message-id field as described in [RFC2822]. The property SHOULD be present on all e-mail messages. See [MS-OXCMAIL] for more information.

2.2.1.11 **PidTagInReplyToId**

Type: PtypString

Corresponds to the in-reply-to field as described in [RFC2822] and containing the original message's PidTagInternetMessageId value. The property MUST be set on a message replies.

2.2.1.12 **PidTagMessageClass**

Type: PtypString

Contains the object type classification. This property MUST be set to "`IPM.Note`" on e-mail objects. The value of PidTagMessageClass for report objects is specified in section 2.2.2 of this document.

2.2.1.13 **PidTagMessageToMe**

Type: PtypBoolean

Indicates that the receiving mailbox owner is one of the primary recipients of this e-mail. This property MAY be absent; if so, the default value of 0x00 is used. If the property is present, it MUST be set to either 0x01, in which case, the receiving mailbox owner is specifically named as a primary recipent of this e-mail and is not part of a distribution list; or 0x00, in which case the receiving mailbox owner is not a primary recipient of this e-mail.

2.2.1.14 **PidTagMessageCcMe**

Type: PtypBoolean

Indicates that the receiving mailbox owner is a carbon copy recipient of this e-mail. This property MAY be absent; if so, the default value of 0x00 is used. If the property is present, it MUST be set to either 0x01, in which case, the receiving mailbox owner is specifically named as a carbon copy recipent of this e-mail and is not part of a distribution list; or 0x00, in which case, the receiving mailbox owner is not a carbon copy recipient of this e-mail.

2.2.1.15 **PidTagMessageRecipientMe**

Type: PtypBoolean

Indicates that the receiving mailbox owner is a primary or a carbon copy recipient of this e-mail. This property MAY be absent; if so, the default value of 0x00 is used. If the property is present, it MUST be set to either 0x01, in which case, the receiving mailbox owner is specifically named as a primary or a carbon copy recipent of this e-mail and is not part of a distribution list; or 0x00 in which case the receiving mailbox owner is not a primary and not a carbon copy recipient of this e-mail.

2.2.1.16 **PidTagOriginatorDeliveryReportRequested**

Type: PtypBoolean

Indicates whether an e-mail sender requests an e-mail delivery receipt from the messaging system. This property MUST be set to either 0x01, in which case, the sender requests the delivery report be sent to the e-mail sender or designated report receiver when the e-mail is delivered; or 0x00 if the e-mail sender does not want to receive the delivery receipt.

2.2.1.17 **PidTagOriginatorNonDeliveryReportRequested**

Type: PtypBoolean

Specifies whether an e-mail sender requests suppression of nondelivery receipts. This property MAY be absent, if so, the server automatically generates and sends a nondelivery receipt to the e-mail sender. If this property is present, it MUST be set to either 0x00, in which case the e-mail sender requests suppression of nondelivery receipt; or 0x01, in which case the nondelivery receipt is generated and sent.

2.2.1.18 **PidTagOriginalSensitivity**

Type: PtypInteger32

Contains the sensitivity value of the original e-mail message. The property MUST be set on the replying and forwarding e-mail messages to the value of the PidTagSensitivity as specified in [MS-OXCMSG] of the original e-mail message.

2.2.1.19 **PidTagReceivedRepresentingAddressType**

Type: PtypString

Contains the e-mail address type, as specified in [MS-OXCDATA] recipient row AddressType and section 2.2.4.3 of this document, for the end user represented by the receiving mailbox owner . If the receiving mailbox owner receives the e-mail on his or her own behalf, this property MUST be set to the value of PidTagReceivedByAddressType.

2.2.1.20 **PidTagReceivedRepresentingEmailAddress**

Type: PtypString

Contains the e-mail address, as specified in [MS-OXCDATA] recipient row E-mailAddress or ExchangeAddress, for the end user represented by the receiving mailbox owner. If the receiving mailbox owner receives the e-mail on his or her own behalf, this property MUST be set to the value of PidTagReceivedByEmailAddress.

2.2.1.21 **PidTagReceivedRepresentingEntryId**

Type: PtypBinary

An address book entry ID containing an identifier, as specified in [MS-OXCDATA] address book EntryId, of the end user represented by the receiving mailbox owner. If the receiving

mailbox owner receives the e-mail on his or her own behalf, this property MUST be set to the value of PidTagReceivedByEntryId.

2.2.1.22 **PidTagReceivedRepresentingName**

Type: PtypString

Contains the display name, as specified in [MS-OXCDATA] recipient row DisplayName, for the end user represented by the receiving mailbox owner. If the receiving mailbox owner receives the e-mail on his or her own behalf, this property MUST be set to the value of PidTagReceivedByName.

2.2.1.23 **PidTagReceivedRepresentingSearchKey**

Type: PtypBinary

An address book SearchKey containing a binary-comparable key, as specified in [MS-OXCDATA] recipient row SearchKey, of the end user represented by the receiving mailbox owner. This property is computed in the same way that PidTagReceivedBySearchKey is computed. If the receiving mailbox owner receives the e-mail on his or her own behalf, this property MUST be set to a value identical to the value of PidTagReceivedBySearchKey.

2.2.1.24 **PidTagReadReceiptRequested**

Type: PtypBoolean

Specifies whether the e-mail sender requests an read receipt from all recipients when this e-mail is read or opened. This property MAY be absent, in which case, no read receipt is sent to the e-mail's sender. If the property is present, it MUST be set to either 0x01, in which case an e-mail's sender requests the read receipt from the messaging system; or 0x00 in which case no read receipt is requested by an e-mail's sender.

If an e-mail object, with PidTagReadReceiptRequested set to 0x01, is deleted as a deletion operation defined in [MS-OXCFOLD], or expires due to the time limit (see section 2.2.3.7 PidTagExpiryTime) before the read receipt for this e-mail is generated, a nonread receipt is generated and sent to the e-mail's sender or designated receipt recipient.

2.2.1.25 **PidTagReceivedByAddressType**

Type: PtypString

MUST contain the e-mail message receiver's e-mail address type as specified in [MS-OXCDATA] recipient row AddressType and section 2.2.4.3 of this document.

2.2.1.26 **PidTagReceivedByEmailAddress**

Type: PtypString

MUST contain the e-mail message receiver's e-mail address as specified in [MS-OXCDATA] recipient row EmailAddress or ExchangeAddress.

2.2.1.27 **PidTagReceivedByEntryId**

Type: PtypBinary

An address book entry ID property that MUST contain the e-mail message receiver of the e-mail object. The address book entry ID data format is specified in [MS-OXCDATA] recipient row Entry IDs.

2.2.1.28 **PidTagReceivedByName**

Type: PtypString

MUST contain the e-mail message receiver's display name, as specified in [MS-OXCDATA] recipient row DisplayName.

2.2.1.29 **PidTagReceivedBySearchKey**

Type: PtypBinary

An address book SearchKey property that contains a binary-comparable key used to identify correlated objects for a search. This property MUST be computed and set by concatenating the message receiver's AddressType and EmailAddress with a colon in between , (for example,<TYPE>:<E-MAIL ADDRESS>) as specified in [MS-OXOABK] and [MS-OXCDATA] recipient SearchKey.

2.2.1.30 **PidTagRecipientReassignmentProhibited**

Type: PtypBoolean

Specifies whether adding additional or different recipients, as with forwarding the message, is prohibited for the e-mail message. This property is set based on the e-mail message's PidTagSensitivity value as specified in [MS-OXCMSG]. If PidTagSensitivity is set to 0x00000000 (normal) or 0x00000003 (confidential), this property MUST be set to 0x00 or absent meaning that adding additional or different recipients to the e-mail message is allowed. If the e-mail object's PidTagSensitivity is set to 0x00000001(personal) or 0x00000002 (private), this property MUST be set 0x01 to prevent adding additional or different recipients of this e-mail through forwarding.

2.2.1.31 **PidTagReplyRecipientEntries**

Type: PtypBinary

A FLATENTRYLIST of address book entry IDs for recipients that are to get a reply. When PidTagReplyRecipientEntries and PidTagReplyRecipientNames are defined, the reply is sent to all of the recipients identified by these two properties. This property MAY be absent, in which case, a reply is sent only to the user identified by PidTagSenderEntryId. If present, the property MUST be set to a FLATENTRYLIST of recipient EntryIds as specified in [MS-OXCDATA].

PidTagReplyRecipientEntries and PidTagReplyRecipientNames properties MUST be set in a way that they contain the same number of recipients in the same order.

2.2.1.32 **PidTagReplyRecipientNames**

Type: PtypString

Contains a list of display names for recipients that are to get a reply. The property MAY be absent, in which case, a reply is sent only to the user identified by PidTagSenderName. If present, the property MUST be set to one string, with the address book entry's recipient display names separated by semicolons as specified in [MS-OXCDATA].

2.2.1.33 **PidTagReplyRequested**

Type: PtypBoolean

Specifies whether a reply to the e-mail is requested by the e-mail's sender. The property MAY be absent, in which case, the reply to the e-mail message is not requested. If the property is present, it MUST be set to either 0x01 if an e-mail sender requests a reply to the e-mail from recipients; or 0x00 which is the same handling as if the property is absent.

2.2.1.34 **PidTagResponseRequested**

Type: PtypBoolean

Specifies whether an e-mail sender requests a response to a meeting request as specified in [MS-OXOCAL] or a voting response (see section 2.2.1.60). This property MAY be absent; if so, the default value 0x00 is used. If present, it MUST be set to either 0x01, in which case, the response to the e-mail message is requested; or 0x00 in which case, the response to the e-mail is not requested.

2.2.1.35 **PidTagSendRichInfo**

Type: PtypBoolean

Specifies whether the sender can receive all message content, including RTF and OLE objects. This property MAY be absent, in which case the default value of 0x00 is used. If the property is present, this property MUST be set to either 0x01 indicating that the sender can receive all message contents, or 0x00 which indicates that the sender of the e-mail message is using a different type of e-mail client.

2.2.1.36 **PidTagSenderAddressType**

Type: PtypString

MUST contain the sending mailbox owner's e-mail address type as specified in the [MS-OXCDATA] Recipient Row section and section 2.2.4.3 of this document.

2.2.1.37 **PidTagSenderEmailAddress**

Type: PtypString

MUST contain the sending mailbox owner's e-mail address as specified in [MS-OXCDATA] Recipient Row EmailAddress or Exchange Address.

2.2.1.38 **PidTagSenderEntryId**

Type: PtypBinary

An address book entry ID that MUST contain the sending mailbox owner's address book entry ID as specified in [MS-OXCDATA] Address Book Entry ID.

2.2.1.39 **PidTagSenderName**

Type: PtypString

MUST contain the sending mailbox owner's display name as specified in [MS-OXCDATA] Recipient Row DisplayName.

2.2.1.40 **PidTagSenderSearchKey**

Type: PtypBinary

An address book SearchKey property that MUST contain a binary-comparable key computed by concatenating sending mailbox owner's PidTagAddressType and PidTagEmailAddress with a colon in between (for example, <TYPE>:<E_MAIL ADDRESS>) as specified in [MS-OXOABK] and [MS-OXCDATA] Recipient Row.

2.2.1.41 **PidTagSentRepresentingAddressType**

Type: PtypString

Contains an e-mail address type (see section 2.2.4.3 of this document) for the end user represented by the sending mailbox owner. If the sending mailbox owner is sending on his or her own behalf, this property MUST be set to the value of PidTagSenderAddressType.

2.2.1.42 **PidTagSentRepresentingEmailAddress**

Type: PtypString

Contains an e-mail address, as specified in [MS-OXCDATA] Recipient Row EmailAddress or ExchangeAddress, for the end user represented by the sending mailbox owner. If a sending mailbox owner is sending on his or her own behalf, this property MUST be set to the value of PidTagSenderEmailAddress.

2.2.1.43 **PidTagSentRepresentingEntryId**

Type: PtypBinary

An address book entry ID, as specified in [MS-OXCDATA] Address Book Entry ID, that contains the identifier of the end user represented by the sending mailbox owner. If a sending mailbox owner is sending on his or her own behalf, this property MUST be set to the value of PidTagSenderEntryId.

2.2.1.44 **PidTagSentRepresentingName**

Type: PtypString

Contains the display name for the end user represented by the sending mailbox owner. If a sending mailbox owner is sending on his or her own behalf, this property MUST be set to the value of PidTagSenderName.

2.2.1.45 **PidTagSentRepresentingSearchKey**

Type: PtypBinary

An address book SearchKey, as specified in [MS-OXCDATA] Recipient Row, containing a binary-comparable key which represents the end user represented by the sending mailbox owner. If a sending mailbox owner sends on his or her own behalf, this property MUST be set to the value of PidTagSenderSearchKey.

2.2.1.46 **PidTagSubjectPrefix**

Type: PtypString

Specified in [MS-OXCMSG]. On an e-mail object, this property typically represents an action on the e-mail message, such as "RE: " for replying and "FW: " for forwarding. This property MAY be absent in which case, there is no subject prefix for the e-mail message.

2.2.1.47 **PidTagTransportMessageHeaders**

Type: PtypString

Contains transport specific message envelope information for e-mail, as specified in [RFC2822]. For outgoing messages with recipients who have a Simple Mail Transfer Protocol (SMTP) address type, the client MUST set this property, and for incoming messages from a sender who has an SMTP address type, the server MUST set this property to a copy of the beginning of the message stream as received from SMTP, up to the first blank line (double **CRLF**).

2.2.1.48 **PidLidInternetAccountName**

Type: PtypString

Specifies the user-visible e-mail account name through which the e-mail messages is sent. The format of this string is implementation dependent. This property can be used by the client to determine which server to direct the mail to, but is optional and the value has no meaning to the server.

2.2.1.49 **PidLidInternetAccountStamp**

Type: PtypString

Specifies the e-mail account ID through which the e-mail message is sent. The format of this string is implementation dependent. This property can be used by the client to determine which server to direct the mail to, but is optional and the value has no meaning to the server.

2.2.1.50 **PidTagPrimarySendAccount**

Type: PtypString

Specifies the first server that a client SHOULD attempt to send the mail with. The format of this property is implementation dependent. This property can be used by the client to determine which server to direct the mail to, but is optional and the value has no meaning to the server.

2.2.1.51 **PidTagNextSendAcct**

Type: PtypString

Specifies the server that a client is currently attempting to use to send a mail. The format of this property is implementation dependent. This property can be used by the client to determine which server to direct the mail to, but is optional and the value has no meaning to the server.

2.2.1.52 **PidLidUseTnef**

Type: PtypBoolean

Specifies whether **Transport Neutral Encapsulation Format (TNEF)** SHOULD be included on a message when the message is converted from TNEF to **Multipurpose Internet Mail Extensions (MIME)** or SMTP format. This property MAY be absent, and if so, implementers of this protocol MUST NOT include TNEF on the message.

2.2.1.53 **Attachments**

The client MAY use Attachments as specified in [MS-OXCMSG].

2.2.1.54 **Categories**

The client MAY set categories on an e-mail message as specified in [MS-OXCMSG].

2.2.1.55 **Contacts**

The client MAY set the contacts on e-mail message as specified in [MS-OXOCNTC] and [MS-OXCMSG].

2.2.1.56 **Flags**

The client MAY set flags as specified in [MS-OXOFLAG].

2.2.1.57 **Reminders**

The client MAY set reminders as specified in [MS-OXORMDR].

2.2.1.58 **Recipients**

The client MUST add recipients to an e-mail message using **RopAddRecipients** as specified in [MS-OXCMSG]. PidTagRecipientType MUST be set to 0x00000001 for the Primary

recipients, 0x00000002 for carbon copy recipients, or 0x00000003 for blind carbon copy recipients. For complete recipient row definition, see [MS-OXCDATA] Recipient Row section.

2.2.1.59 **PidLidAutoProcessState**

Type: PtypInteger32

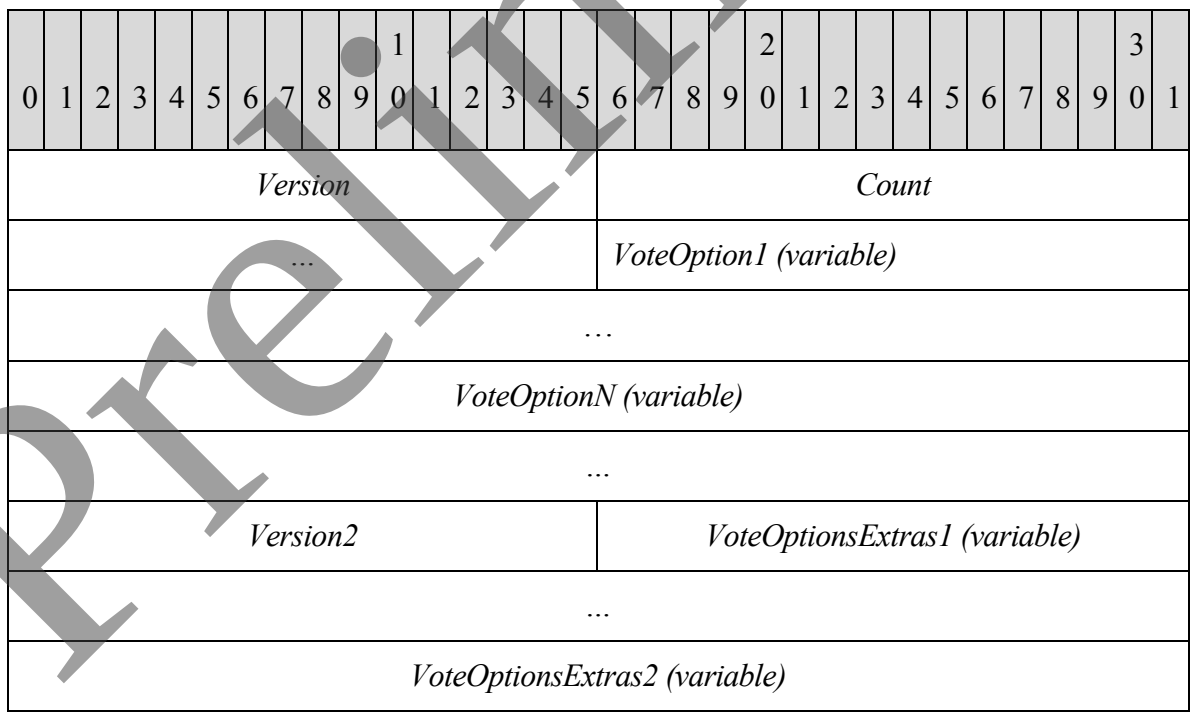
Specifies the options used in automatic processing of e-mail messages. The property MAY be absent, in which case the default value of 0x00000000 is used. If set, this property MUST be set to one of the values below.

Value	Description
0x00000000	Don't auto-process the message.
0x00000001	Process the message automatically or when the message is opened.
0x00000002	Process when the message is opened only.

2.2.1.60 **PidLidVerbStream**

Type: PtypBinary

Specifies what voting responses the user can make in response to the message. Client processing of this stream (or the lack of this stream) is described in section 3.1.5. The format is described below.



…

Version (WORD): MUST be 0x0102

Count (**DWORD**): Specifies the number of VoteOption and VoteOptionExtras to follow.

VoteOption1 (variable length structure): The first VoteOption structure described in section 2.2.1.61.1.

VoteOptionN (variable length structure): The last VoteOption structure described in section 2.2.1.61.1.

Version2 (WORD): MUST be 0x0104.

VoteOptionExtras1 (variable length structure): The first VoteOptionExtras structure described in section 2.2.1.61.2.

VoteOptionExtrasN (variable length structure): The last VoteOptionExtras structure described in section 2.2.1.61.2.

2.2.1.60.1 VoteOption Structure

The verb stream contains two parallel arrays of **VoteOption** and **VoteOptionExtra** structures. Each element in these two arrays, when combined, describe a single voting option which can be taken by the user in response to the message. The format of the **VoteOption** structure is described below.

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>20</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>30</th><th>1</th></tr>
<tr><td colspan="32">VerbType</td></tr>
<tr><td colspan="8">DisplayNameCount</td><td colspan="24">DisplayName (variable)</td></tr>
<tr><td colspan="32">…</td></tr>
<tr><td colspan="8">MsgClsNameCount</td><td colspan="24">MsgClsName (variable)</td></tr>
<tr><td colspan="32">…</td></tr>
<tr><td colspan="8">Internal1StringCount</td><td colspan="8">DisplayNameCountRepeat</td><td colspan="16">DisplayNameRepeat (variable)</td></tr>
<tr><td colspan="32">…</td></tr>
<tr><td colspan="32">Internal2</td></tr>
</table>

Internal3	*fUseUSHeaders*
...	*Internal4*
...	*SendBehavior*
...	*Internal5*
...	*ID*
...	*Internal6*
...	

VerbType (DWORD): MUST be 4 (0x00000004).

DisplayNameCount (1 byte): Count of characters in the following string.

DisplayName [ANSI String (NOT null terminated)]: The localized display name of the Voting option (for example, "Yes"), without a null terminator.

MsgClsNameCount (1 byte): Count of characters in the following string. MUST be 8 (0x08).

MsgClsName [ANSI String (NOT null terminated)]: MUST be "IPM.Note", without a null terminator.

Internal1StringCount (1 byte): Count of characters in the following string. MUST be 0x00 for voting options.

Internal1String [ANSI String (NOT null terminated)]: MUST not be present, as Internal1StringCount is always 0x00 for a voting option.

DisplayNameCountRepeat (1 byte): MUST be the same as DisplayNameCount.

DisplayNameRepeat [ANSI String (NOT null terminated)]: MUST be the same as DisplayName.

Internal2 (DWORD): MUST be 0 (0x00000000).

Internal3 (1 byte): MUST be 0x00.

fUseUSHeaders (DWORD): Indicates that a US style reply header is to be used in the response message (as opposed to a localized response header). The value MUST be either 0x00000001, using US style reply header, or 0x00000000 otherwise.

Internal4 (DWORD): MUST be 0x00000001.

SendBehavior (DWORD): Behavior on send. When a user chooses a voting option, SendBehavior specifies if the user is to be prompted to edit the response mail, or automatically send it on behalf of the user. The value of this field MUST be one of the values defined in the following table.

Value	Description
0x00000001	Automatically send the voting response message.
0x00000002	Prompt the user if he or she would like to automatically send or edit the voting response first..

Internal5 (DWORD): MUST be 2 (0x00000002).

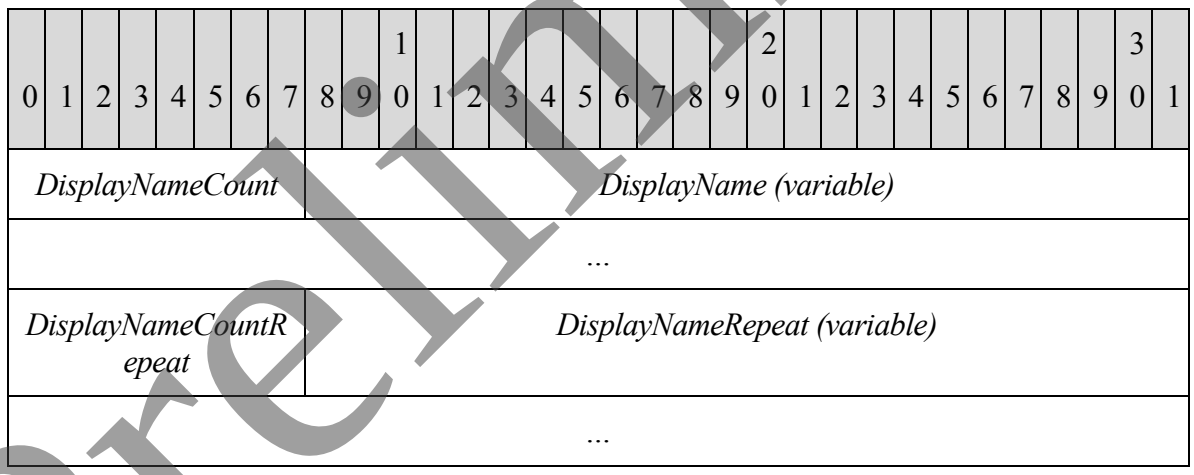
ID (DWORD): Specifies a numeric identifier for this voting option. The client SHOULD specify 1 for the first VoteOption, and monotomically increase this value for each subsequent VoteOptions.

Internal6 (DWORD): MUST be -1 (0xFFFFFFFF).

Note that because the DisplayNameCount (and DisplayNameCountRepeat) fields are 1-byte long and contain the count of characters in DisplayName (and DisplayNameRepeat), this implies a length limit of 255 characters in the DisplayName of any voting option.

2.2.1.60.2 VoteOptionExtras Structure

Each element contains additional information about the corresponding *VoteOptions* entry. The format is described below.



DisplayNameCount (1 byte): Count of Unicode characters (NOT bytes) in the following string.

DisplayName [Unicode String (NOT null terminated)]: The display name of this voting option, as a Unicode string, without a null terminator.

DisplayNameCountRepeat (1 byte): Count of characters in the following string. MUST be the same as DisplayNameCount.

DisplayNameRepeat [Unicode String (NOT null terminated)]: MUST be the same as DisplayName.

2.2.1.61 **PidLidVerbResponse**

Type: PtypString

Specifies the voting option a respondent has selected. This property SHOULD be set to one of the voting button display names on which the respondent votes.

2.2.1.62 **PidTagTargetEntryId**

Type: PtypBinary

Used in conjunction with an optimizing send client. See sections 3.1.4.4 and 3.2.5.1.2.8

2.2.2 Message Status Reports

2.2.2.1 **PidTagMessageClass**

Type: PtypString

Contains a message object class name. For report messages, the property MUST be set to the value in the form: "`REPORT.X.<receipt types>`" where X is the original message class name, such as "`IPM.NOTE`" for an e-mail object and <receipt-type> is one of the following receipt types:

IPNRN: Read receipt

IPNNRN: Non-read receipt

DR: Delivery receipt

NDR: Non-delivery receipt

Therefore, the report messages of the IPM.NOTE message class name are:

Report type	Message class name (PtypString)
Read Receipt	`REPORT.IPM.NOTE.IPNRN`
Nonread Receipt	`REPORT.IPM.NOTE.IPNNRN`
Delivery Receipt	`REPORT.IPM.NOTE.DR`
Nondelivery Receipt	`REPORT.IPM.NOTE.NDR`

2.2.2.2 **PidTagOriginalDeliveryTime**

Type: PtypTime

MUST be set on read/nonread report objects or replying/forwarding message objects with the value from PidTagMessageDeliveryTime of the original message.

2.2.2.3 **PidTagOriginalDisplayTo**

Type: PtypString

MUST be set on report messages to the value of PidTagDisplayTo from the original message, if present.

2.2.2.4 **PidTagOriginalDisplayCc**

Type: PtypString

MUST be set on report messages to the value of PidTagDisplayCc from the original message, if present.

2.2.2.5 **PidTagOriginalDisplayBcc**

Type: PtypString

MUST be set on report messages to a copy of PidTagDisplayBcc from the original message, if present.

2.2.2.6 **PidTagOriginalSenderAddressType**

Type: PtypString

MUST be set on delivery report messages to the value of the original message sender's PidTagSenderAddressType as specified in [MS-OXCDATA] Recipient Row.

2.2.2.7 **PidTagOriginalSenderEmailAddress**

Type: PtypString

MUST be set on delivery report messages to the value of the original message sender's PidTagSenderEmailAddress as specified in section 2.2.1.37.

2.2.2.8 **PidTagOriginalSenderEntryId**

Type: PtypBinary

This address book entry ID property MUST be set on delivery report messages to the value of PidTagSenderEntryId from the original e-mail as specified in section 2.2.1.38.

2.2.2.9 **PidTagOriginalSenderName**

Type: PtypString

MUST be set on delivery report messages to the value of the original message sender's PidTagSenderName as specified in section 2.2.1.39.

2.2.2.10 **PidTagOriginalSenderSearchKey**

Type: PtypBinary

This address book Search Key property MUST be set on delivery report messages to the value of PidTagSenderSearchKey of the original e-mail message, as specified in section 2.2.1.40.

2.2.2.11 **PidTagOriginalSentRepresentingAddressType**

Type: PtypString

Contains the address type of the end user represented by the original e-mail message sender. It MUST be set to the value of PidTagSentRepresentingAddressType of the original e-mail message as specified in section 2.2.1.41.

2.2.2.12 **PidTagOriginalSentRepresentingEmailAddress**

Type: PtypString

Contains the e-mail address of the end user represented by the original e-mail message sender. It MUST be set to the value of PidTagSentRepresentingEmailAddress of the original e-mail message as specified in section 2.2.1.42.

2.2.2.13 **PidTagOriginalSentRepresentingEntryId**

Type: PtypBinary

An address book entry ID property containing the entry identifier of the end user represented by the original message sender. It MUST be set to the value of PidTagSentRepresentingEntryId property of the original message as specified in section 2.2.1.43.

2.2.2.14 **PidTagOriginalSentRepresentingName**

Type: PtypString

Contains the display name of the end user represented by the original e-mail message sender; MUST be set to the value of PidTagSentRepresentingName of the original e-mail message as specified in section 2.2.1.44.

2.2.2.15 **PidTagOriginalSentRepresentingSearchKey**

Type: PtypBinary

An address book SearchKey property containing the SearchKey of the end user represented by the original message sender. It MUST be set to the value of PidTagSentRepresentingSearchKey property of the original message as specified in section 2.2.1.45.

2.2.2.16 **PidTagOriginalSubject**

Type: PtypString

Specifies the subject of the original message and MUST be set to the concatenated values of the PidTagSubjectPrefix and PidTagNormalizedSubject properties of the original message.

2.2.2.17 **PidTagOriginalSubmitTime**

Type: PtypTime

Specifies the original e-mail's submission date and time and MUST be set to the value of PidTagClientSubmitTime. The property is used in reports only and once set, it MUST NOT be changed.

2.2.2.18 **PidTagParentKey**

Type: PtypBinary

Contains the search key used to correlate the original message and the reports about the original message. The server MUST set the property on the report message to the value of PidTagSearchKey of the original e-mail message, as specified in [MS-OXCMSG].

2.2.2.19 **PidTagReportTag**

Type: PtypBinary

Contains the data used to correlate the report and the original message. The property MAY be absent if the sender does not request a reply or response for the original e-mail. If the original e-mail object has either PidTagResponseRequested set to 0x01 or PidTagReplyRequested to 0x01, the property MUST be set on the original e-mail object which follows the format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Cookie																															
Continue Cookie																															
0x00								*Version*																							
Continue version								*StoreEntryIdSize*																							
Continue size								*StoreEntryId (variable)*																							
FolderEntryIdSize																															
FolderEntryId (variable)																															
MessageEntryIdSize																															
MessageEntryId (variable)																															
SearchFolderEntryIdSize																															

SearchFolderEntryId (variable)
MessageSearchKeySize
MessageSearchKey(variable)
Ansi Text Size
Ansit Text (variable)

Cookie (9 bytes: string): Nine characters used for validation; MUST be "PCDFEB09".

Version (4 bytes): MUST be either CurrentVersion (0x00010002) or NoSearchFolderVersion (0x00010001).

StoreEntryIdSize (4 bytes): size of StoreEntryId.

StoreEntryId (variable length of bytes): If the size is 0x00000000, this field MUST be omitted. If the size is not zero, this field MUST be filled with the specified number of bytes.

FoldeEntryIdSize (**4 bytes)**: size of FolderEntryId.

FolderEntryId (variable length of bytes): If the size is 0x00000000, this field MUST be omitted. If the size is not zero, the field MUST be filled with the specified number of bytes.

MessageEntryIdSize (4 bytes): size of MessageEntryId.

MessageEntryId (variable length of bytes): If the size is 0x00000000, this field MUST be omitted. If the size is not zero, the field MUST be filled with the specified number of bytes.

SearchFolderEntryIdSize (4 bytes): If Version equals to the CurrentVersion, this MUST be the real size of SearchFolderEntryId. Otherwise, MUST be set to 0x00000000.

SearchFolderEntryId (variable length of bytes): If size is not zero, this field MUST be the specified number of bytes of SearchFolderEntryId. Otherwise, if size is zero, this field MUST be omitted.

MessageSearchKeySize (4 bytes): Size of MessageSearchKey.

MessageSearchKey (variable length of bytes): MUST be set to the specified number of bytes.

Ansi Text Size (4 bytes): Number of chars in the next field.

Ansi Text (variable bytes): Specified number of chars.

2.2.2.20 **PidTagReportText**

Type: PtypString

Contains the optional text for a report message. The property MAY be absent, in which case, there is no report text from the server. If present, a server MUST set this property to the text string in response to a report type from the underlying messaging system.

2.2.2.21 **PidTagReadReceiptEntryId**

Type: PtypBinary

An address book entry ID property, as specified in [MS-OXCDATA], representing the user to whom a read receipt is directed. This property is only used and validated if PidTagReadReceiptRequested is set to 0x01. The property MAY be absent, in which case, PidTagReportEntryId is used as an alternative. If neither property is present, PigTagSenderEntryId is used as the user who receives the read receipt.

2.2.2.22 **PidTagReadReceiptSearchKey**

Type: PtypBinary

An address book SearchKey property, as specified in [MS-OXCDATA], representing the user to whom a read receipt is directed. This property is only used and validated if PidTagReadReceiptRequested is set to 0x01. The property MAY be absent, in which case, PidTagReportSearchKey is used as an alternative. If neither properties is present, PigTagSenderSearchKey is used as the user who receives the read receipt.

2.2.2.23 **PidTagSubjectPrefix**

Described in section 2.2.1.46<3>.

2.2.3 E-mail Submission Properties

The following properties are specified in [MS-OXPROPS], and are properties of recipients in the recipient table. They are specially handled during message submission.

2.2.3.1 **PidTagRecipientType**

Type: PtypInteger32

Represents recipient type of a recipient on the message that MUST be set on each recipient. The format of this property is below.

Value	Description
0x00000000	The recipient is the message originator.
0x00000001	The recipient is a primary recipient.
0x00000002	The recipient is a carbon copy recipient
0x00000003	The recipient is a blind carbon copy recipient.

Additionally, the following flags can be combined with the above values:

Flag	Description
0x10000000	If a message failed to be delivered to some recipients, the client can mark the message as a **resend message** by setting the mfResend bit (0x00000080) in the PidTagMessageFlags property. Combining with the value of PidTagRecipientType, indicates that the server MUST resend the message to the recipient.
0x80000000	On a resend message, the recipient received the message successfully and does not need to receive it again. The server MUST NOT send the resend message to the recipient.

2.2.3.2 **PidTagDeferredSendNumber**

Type: PtypInteger32

When sending a message is deferred, PidTagDeferredSendNumber SHOULD be set along with PidTagDeferredSendUnits if PidTagDeferredSendTime is absent. The value MUST be set between 0x00000000 and 0x000003E7 (0 and 999).

PidTagDeferredSendNumber is used for computing PidTagDeferredSendTime when PidTagDeferredSendTime is not present. Also see section 2.2.3.3 PidTagDeferredSendUnits and 2.2.3.4 PidTagDeferredSendTime.

2.2.3.3 **PidTagDeferredSendUnits**

Type: PtypInteger32

Specifies the unit of time that PidTagDeferredSendNumber SHOULD be multiplied by. Also see section 2.2.3.4 PidTagDeferredSendTime. If set, PidTagDeferredSendUnits MUST be one of the following values:

PidTagDeferredSendUnits	Meaning (time of)
0x00000000	Minutes, for example 60 seconds
0x00000001	Hours, for example 60x60 seconds
0x00000002	Day, for example 24x60x60 seconds
0x00000003	Week, for example 7x24x60x60 seconds

2.2.3.4 **PidTagDeferredSendTime**

Type: PtypTime

MAY be present if a client would like to defer sending the message after a certain amount of time.

If PidTagDeferredSendUnits and PidTagDeferredSendNumber are present, the value of PidTagDeferredSendTime is recomputed using the following formula and the old value is ignored.

```
PidTagDeferredSendTime = PidTagClientSubmitTime +
                         PidTagDeferredSendNumber *
                         TimeOf(PidTagDeferredSendUnits)
```

If PidTagDeferredSendTime value is earlier than current time (in UTC), the message is sent immediately.

2.2.3.5 **PidTagExpiryNumber**

Type: PtypInteger32

Used along with PidTagExpiryUnits to define the expiry send time. If PidTagExpiryNumber is present, the value MUST be set between 0x0000000 and 0x000003E7 (0 and 999).

2.2.3.6 **PidTagExpiryUnits**

Type: PtypInteger32

Used to describe the unit of time that PidTagExpiryNumber multiplies. If set, PidTagExpiryUnits MUST be one of the following values:

PidTagExpiryUnits	Meaning (TimeOf)
0x00000000	Minutes, for example 60 seconds
0x00000001	Hours, for example 60x60 seconds
0x00000002	Day, for example 24x60x60 seconds
0x00000003	Week, for example 7x24x60x60 seconds

2.2.3.7 **PidTagExpiryTime**

Type: PtypTime

MAY be present when a client would like to receive an expiry event if the message arrives late.

If PidTagExpiryNumber and PidTagExpiryUnits are present, the value of PidTagExpiryTime is recomputed by the following formula and the old value is ignored.

```
PidTagExpiryTime = PidTagClientSubmitTime +
                   PidTagExpiryNumber *
                   TimeOf(PidTagExpiryUnits)
```

2.2.3.8 **PidTagDeleteAfterSubmit**

Type: PtypBoolean

Indicates that the original message MUST be deleted after the message is sent. If the property is not present, the server uses the value is 0x00.

2.2.3.9 **PidTagMessageDeliveryTime**

Type: PtypTime

MUST be set to current time in UTC by the server upon receiving a message.

2.2.3.10 **PidTagSentMailSvrEID**

Type: PtypBinary

Represents the Sent Mail folder for the message. This folder MUST not be a Search Folder. The server requires write permission on the folder so that the sent e-mail message can be copied to the Sent Mail folder.

If present, a copy of the message MUST be created in the specified folder after the message is sent.

2.2.3.11 **PidTagClientSubmitTime**

Type: PtypTime

MUST be set by the server to current time in UTC when the e-mail message is submitted.

2.2.4 ROPs Used in Sending Message

The format of the **RopSubmitMessage** and **RopAbortSubmit** request and response buffers are specified in the [MS-OXCROPS] protocol.

2.2.4.1 **RopSubmitMessage**

RopSubmitMessage request is sent to submit an e-mail message object for sending.

The messaging client MUST log on as a user with sufficient permissions to write messages because the server needs to modify certain properties (see section 3.2 Server Details).

The message is identified by the handle index which is maintained by both the server and client about the message object. The handle index MUST be acquired by a previous **RopOpenMessage** or **RopCreateMessage** request.

When a message is submitted, any pending changes on the message are saved to the server.

2.2.4.1.1 SubmitFlags

When the messaging client submits the message, this indicates how the message SHOULD be delivered using the following values:

Name	Value	Description
None	0x00	None.
PreProcess	0x01	The message needs to be preprocessed by the server.
NeedsSpooler	0x02	The message is to be processed by a client spooler.

2.2.4.1.2 Return Values

The following table describes the return value in the response buffer.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecShutoffQuotaExceeded	0x000004DD	Indicates that the maximum storage shut-off quota has been exceeded.
ecQuotaExceeded	0x000004D9	Indicates that the storage quota is exceeded for the mailbox, but the user can still receive

		mail.
ecNotSupported	0x80040102	The server object associated with the input handle index in the server object table is not of type Message or the current logon session is a public logon.
ecTooManyRecips	0x00000505	The number of recipients of the message exceed the allowed limit. If this error happens, none of the recipients will receive this message.
ecAccessDenied	0x80070005	The message is an FAI message.
ecRequiresRefResolve	0x0000047E	Attachments contain references to paths that are inaccessible to the server and need to be resolved.

2.2.4.2 **RopAbortSubmit**

Before an e-mail message object is actually processed by the server or a client mail spooler, a messaging client can send **RopAbortSubmit** request with an attempt to abort the submission.

If the operation succeeds, the message currently queued on the server will be removed from the server. Unless the message is submitted for sending again, the message will not be delivered to its recipients.

The message to be aborted submission is identified by the FolderId and MessageId fields in the request buffer. RopSubmitMessage MUST have been invoked upon this message previously.

2.2.4.2.1 Return Values

The following are the values returned in the *ReturnValue* field of the Response Buffer.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecUnableToAbort	0x80040114	The operation cannot be aborted.
ecNotInQueue	0x80040601	The message is no longer in the message store's spooler queue.
ecNotSupported	0x80040102	The server object associated with the input handle index in the server object table is not of type logon or

		the current logon session is a public logon.
ecNotFound	0x8004010F	The parent Folder ID or Message ID is invalid.

2.2.4.3 RopGetAddressTypes

RopGetAddressTypes request is sent by a client to retrieve the address types of recipients supported by server.

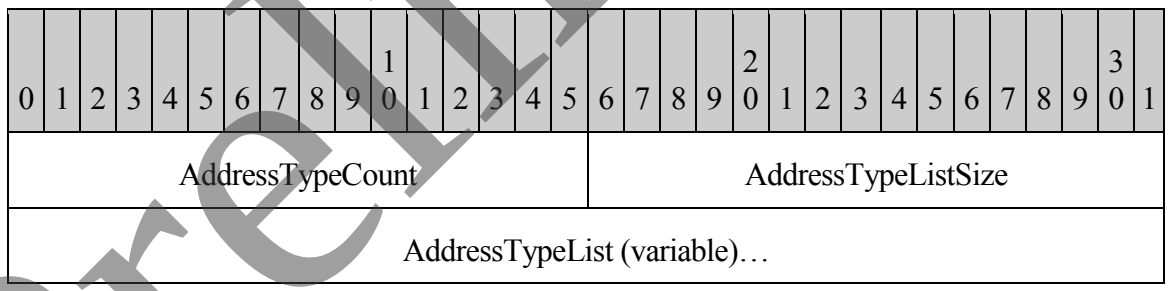
In the request, the server object associated with the input handle index in the server object table is ignored by the server.

2.2.4.3.1 Return Values

The following table describes the return value in the response buffer.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecBufferTooSmall	0x0000047D	The response buffer is not large to hold the results.
ecNotSupported	0x80040102	The server does not support returning address types.

Also in the response buffer, address types are returned in the below format:



AddressTypeCount (WORD): The number of address types returned.

AddressTypeListSize (WORD): The total length of the AddressTypeList followed.

AddressTypeList (variable): An array of NULL terminated ASCII strings each representing an address type. <4>

2.2.5 E-mail Sending and Delivery ROPs

The following ROP requests can be used by a client if it needs to control the receipt of mail that is not delivered directly to the server, or sending of mail from an e-mail account not supported on the server.

2.2.5.1 **RopSetSpooler**

The **RopSetSpooler** request is sent to inform the server that the client intends to act as a mail spooler. The syntax of the **RopSetSpooler** request and response buffers are specified in the [MS-OXCROPS] protocol. This section specifies the syntax and semantics of various fields that are not fully specified in the Remote Operations (ROP) List and Encoding Protocol (see [MS-OXCROPS]). The server allows multiple clients to act as spoolers.

2.2.5.1.1 Request Buffer

InputHandleIndex: The input handle for this operation is a logon object handle.

2.2.5.1.2 Return Values

The following table describes the return value in the response buffer:

Name	Value	Meaning
ecNone	0x00000000	Success.

2.2.5.2 **RopGetTransportFolder**

The **RopGetTransportFolder** request is sent to retrieve the folder ID (FID) of the transport folder. Outgoing messages can be stored in this folder before a **RopTransportSend** request is issued. The syntax of the **RopGetTransportFolder** request and response buffers are specified in the [MS-OXCROPS] protocol. This section specifies the syntax and semantics of various fields that are not fully specified in the Remote Operations (ROP) List and Encoding Protocol (see [MS-OXCROPS]).

2.2.5.2.1 Request Buffer

InputHandleIndex: The input handle for this operation is a logon object handle.

2.2.5.2.2 Return Values

The following table describes the return value in the response buffer:

Name	Value	Meaning
ecNone	0x00000000	Success.

2.2.5.2.3 Response Buffer

The following fields are returned in the response buffer:

FolderID: Contains the folder ID of the transport folder.

2.2.5.3 RopSpoolerLockMessage

The **RopSpoolerLockMessage** request is sent to lock the specified message for spooling. When a message is locked, the server MUST deny **RopAbortSubmit** requests and other requests to lock or access the message. Once a client makes a successful request to mark the message as locked, it MUST subsequently make a request to mark the message as unlocked or finished. The syntax of the **RopSpoolerLockMessage** request and response buffers are specified in the [MS-OXCROPS] protocol. This section specifies the syntax and semantics of various fields that are not fully specified in the Remote Operations (ROP) List and Encoding Protocol (see [MS-OXCROPS]).

2.2.5.3.1 Request Buffer

InputHandleIndex: The input handle for this operation is a logon object handle.

MessageId: Specifies the message to be locked.

LockState (BYTE): Specifies a status to set on the message. The following table describes the valid values:

Name	Value	Meaning
lstLock	0x00	Mark the message as locked.
lstUnlock	0x01	Mark the message as unlocked.
lstFinished	0x02	Mark the message as ready for processing by the server. The server MUST move or delete the message based on the presence of PidTagSentMailSvrEID and PidTagDeleteAfterSubmit properties on the message (specified in section 2.2.3.8 and 2.2.3.10)

2.2.5.3.2 Return Values

The following table describes the return value in the response buffer.

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNoSupport	0x80040102	The server does not support sent message processing, or if the client is not the spooler.

2.2.5.4 RopTransportSend

The **RopTransportSend** request is used to have the server send an e-mail message to recipients. The message to be sent is identified by the handle index which is maintained by both the server and client. The syntax of the **RopTransportSend** request and response buffers are specified in the [MS-OXCROPS] protocol. This section specifies the syntax and semantics of various fields that are not fully specified in the Remote Operations (ROP) List and Encoding Protocol (see [MS-OXCROPS]).

2.2.5.4.1 Request Buffer

InputHandleIndex: The input handle for this operation is the handle of the message object to be sent.

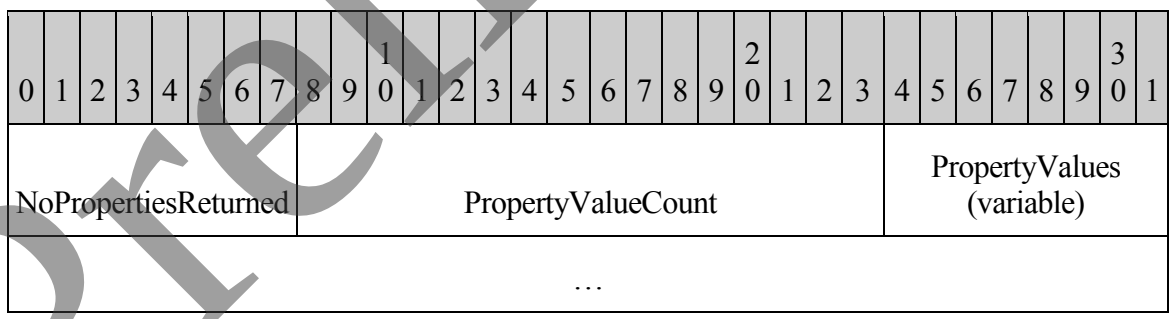
2.2.5.4.2 Return Values

The following table describes the return value in the response buffer:

Name	Value	Meaning
ecNone	0x00000000	Success.
ecNotMe	0x80040502	The server could not handle the message and the message was not sent. The client SHOULD try another server if available.

2.2.5.4.3 Response Buffer

The response buffer contains the following fields:



NoPropertiesReturned (BOOL): 0x00 if properties are returned, 0x01 otherwise.

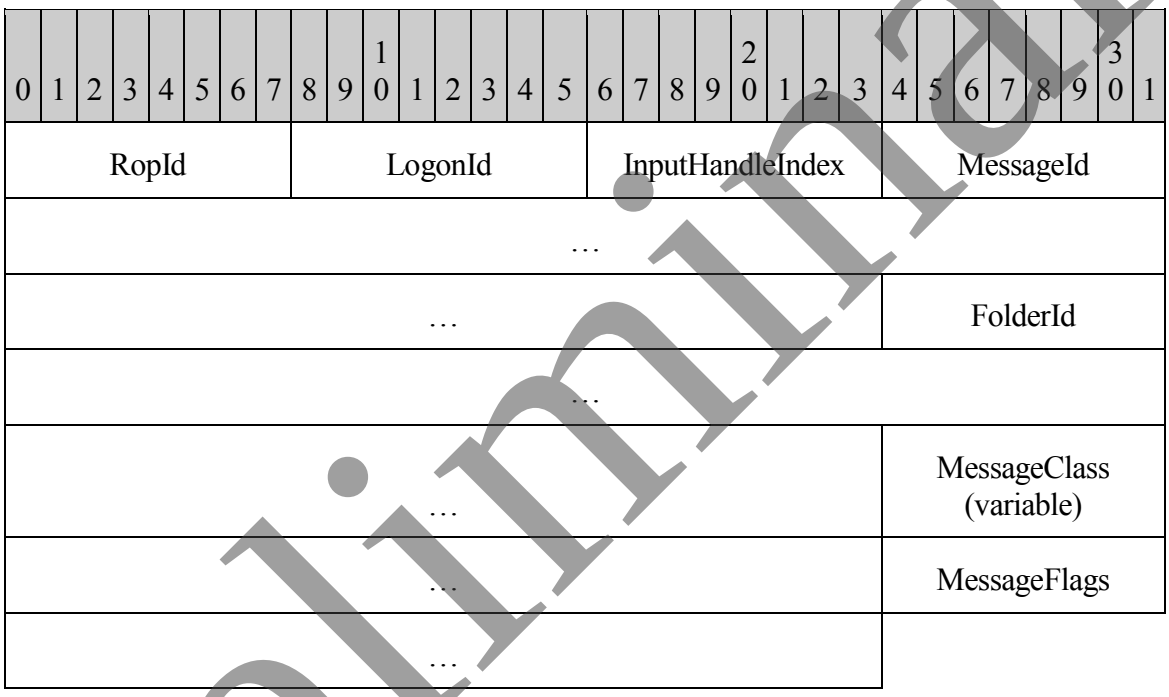
PropertyValueCount (WORD): Number of properties. Only exists if NoPropertiesReturned is 0x00.

PropertyValues (variable): Array of PropertyTag structures. Only exists if NoPropertiesReturned is 0x00. This field MUST contain **PropertyValueCount** tags. The format of PropertyValues is specified in [MS-OXCDATA]. This field contains the properties set on the message by the server in the process of sending the message.

2.2.5.5 RopTransportNewMail

The **RopTransportNewMail** request is used notify the server of new mail delivered to the message store. The syntax of the **RopTransportNewMail** request and response buffers are specified in the [MS-OXCROPS] protocol. This section specifies the syntax and semantics of various fields that are not fully specified in the Remote Operations (ROP) List and Encoding Protocol (see [MS-OXCROPS]).

2.2.5.5.1 Request Buffer



InputHandleIndex: The input handle for this operation is a logon object handle.

MessageId: Specifies the message id of the new message.

FolderId: Specifies the location of the new message.

MessageClass (variable): Zero-terminated ANSI string that specifies the value of PidTagMessageClass of the message.

MessageFlags (DWORD): Specifies the value of PidTagMessageFlags of the message.

2.2.5.5.2 Return Values

The following table describes the return value in the response buffer.

Name	Value	Meaning
ecNone	0x00000000	Success.

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

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

The E-mail Object Protocol abstract data model extends objects specified by other protocols, as listed in the following table:

Object	Protocol
Property Property Bag Messaging Item	[MS-OXPROPS] [MS-OXCMSG]
Messaging User	[MS-OXCDATA] Recipient Row, [MS-OXOABK]
Store	[MS-OXCSTORE]
Folder	[MS-OXCFOLD], [MS-OXOSFOLD]

An e-mail object is a type of property bag, distinguished from other messaging items and property bag types by its default storage location, its message class (the value of its PidTagMessageClass property), and the inclusion of certain sub-objects, as specified in the following sections.

3.1.1.1 **Storage**

An e-mail object is a messaging object with a message class of "IPM.Note". By default, a client implemention stores e-mail items in a folder object which has the container class of "IPF.Note".

From the point of view of the currently logged-on messaging user, an e-mail object is either a Send Note, meaning that the e-mail is to be or has been sent *to* an external messaging user or user agent, or it is a Receive Note, meaning that the e-mail was sent to the current messaging user *from* an external user or user agent.

Within these groupings, an e-mail exists in one of a small number of abstract states, which determines the default storage location for that particular e-mail object, as specified in the following table:

E-mail State	Description	Special Folder
Saved	A Send Note stored within an Inter-Personal Mail (IPM) folder within a store object.	**Drafts folder**
Submitted	A Send Note that is marked to be sent by the server.	**Outbox folder**
Sent	A Send Note that has been claimed by the messaging transport for delivery to another messaging user.	**Sent Mail folder**
Received	A Receive Note that has been placed in the default Receive Folder by the server.	**Inbox folder** (default Receive Folder)

3.1.1.2 Core Objects

The abstract sub-objects which are core to every e-mail object are: Sender, Recipients, Subject, and Body.

3.1.1.2.1 Sender

Message senders are identified by the from properties and the sender properties on an e-mail object. In general, the from properties and the sender properties will identify the same messaging user; for example, the message appears to have been sent by the actual sender of the message. In some cases, however, a message is sent by one user (the actual sender) on behalf of another user (the represented sender). In this case, the from properties identify the represented sender and the sender properties identify the actual sender.

3.1.1.2.1.1 Represented Sender

The represented sender of a message is the messaging user or user agent on whose behalf the message was sent (or will be sent). The from properties associated only with the represented sender are:

- PidTagSentRepresentingAddressType
- PidTagSentRepresentingEmailAddress

- PidTagSentRepresentingEntryId
- PidTagSentRepresentingName
- PidTagSentRepresentingSearchKey
- PidTagOriginalSentRepresentingAddressType
- PidTagOriginalSentRepresentingEmailAddress
- PidTagOriginalSentRepresentingEntryId
- PidTagOriginalSentRepresentingName
- PidTagOriginalSentRepresentingSearchKey

3.1.1.2.1.2 Actual Sender

The actual sender is the owner of the mailbox that sent (or will send) the e-mail. The from properties associated with the actual sender are:

- PidTagSenderAddressType
- PidTagSenderEmailAddress
- PidTagSenderEntryId
- PidTagSenderName
- PidTagSenderSearchKey
- PidTagOriginalSenderAddressType
- PidTagOriginalSenderEmailAddress
- PidTagOriginalSenderEntryId
- PidTagOriginalSenderName
- PidTagOriginalSenderSearchKey

3.1.1.2.2 Recipients

Recipients is a collection of recipients each of which is a messaging user to whom the e-mail is to be (or has been) delivered. As with senders, there are two types of recipients: represented recipients and actual recipients. Within each of these types, there are three subclasses of recipients for an e-mail: To, Carbon Copy (CC), and Blind Carbon Copy (BCC).

3.1.1.2.2.1 Represented Recipients

A represented recipient is the messaging user or user agent on whose behalf the message is being received. The recipient properties associated with represented recipients are:

- PidTagReceivedRepresentingAddressType
- PidTagReceivedRepresentingEmailAddress
- PidTagReceivedRepresentingEntryId
- PidTagReceivedRepresentingName

- PidTagReceivedRepresentingSearchKey

3.1.1.2.2.2 Actual Recipients

An actual recipient is the receiving mailbox owner of a message. The recipient properties associated with actual recipients are:

- PidTagMessageRecipientMe
- PidTagReceivedByAddressType
- PidTagReceivedByEmailAddress
- PidTagReceivedByEntryId
- PidTagReceivedByName
- PidTagReceivedBySearchKey
- PidTagRecipientType

3.1.1.2.2.3 Other From Properties

Another set of from properties are used to identify three subclasses of recipients for an e-mail: To, Carbon Copy (CC), and Blind Carbon Copy (BCC).

The from properties associated with To Recipients are:

- PidTagDisplayTo
- PidTagMessageToMe
- PidTagOriginalDisplayTo

The from properties associated with CC Recipients are:

- PidTagDisplayCc
- PidTagMessageCcMe
- PidTagOriginalDisplayCc

The from properties associated with BCC Recipients are:

- PidTagDisplayBcc
- PidTagOriginalDisplayBcc

3.1.1.2.3 Subject

The Subject is a short text string intended to inform a recipient as to the contents or purpose of the e-mail. The properties associated with the Subject are:

- PidTagNormalizedSubject
- PidTagSubjectPrefix
- PidTagOriginalSubject

3.1.1.2.4 Body

The Body, specified fully by the [MS-OXBBODY] protocol, contains the main contents of the e-mail. The properties associated with the Body are:

- PidTagBlockStatus
- PidTagBody
- PidTagBodyHtml
- PidTagRtfCompressed
- PidTagRtfInSync
- PidTagMessageEditorFormat

3.1.1.3 Other Informational Messaging Properties

Many properties not associated with the preceding core e-mail objects are included with an e-mail in support of other particular sub-objects. These sub-objects, along with their associated properties, are:

- Conversations
    - PidTagConversationIndex
    - PidTagConversationTopic

    If the e-mail message in the conversation thread is given a new subject, this e-mail message starts the new conversation thread with a new PidTagConversationTopic and PidTagConversationIndex.
- Client Options
    - PidTagIconIndex
    - PidTagMessageClass
    - PidTagReadReceiptRequested
    - PidTagReadReceiptEntryId
    - PidTagReadReceiptSearchKey
    - PidTagOriginalSensitivity
    - PidTagRecipientReassignmentProhibited
    - PidTagReplyRequested
    - PidTagResponseRequested
    - PidTagReplyRecipientEntries
    - PidTagReplyRecipientNames
    - PidLidAutoProcessState

- PidLidVerbStream
- PidLidVerbResponse

3.1.1.4 Message Delivery Properties

Many properties are set by the messaging system itself or by a client implementation to control the behavior of the messaging system. These properties are:

- PidTagExpiryTime
- PidTagInternetMessageId
- PidTagOriginatorDeliveryReportRequested
- PidTagOriginatorNonDeliveryReportRequested
- PidTagSendRichInfo
- PidTagTransportMessageHeaders
- PidTagOriginalDeliveryTime
- PidTagOriginalSubmitTime
- PidTagParentKey
- PidTagReportTag
- PidTagReportText
- PidTagMessageFlags
- PidTagMessageDeliveryTime
- PidTagDeferredSendNumber
- PidTagDeferredSendUnits
- PidTagDeferredSendTime
- PidTagExpiryNumber
- PidTagExpiryUnits

3.1.2 Timers

None.

3.1.3 Initialization

A client can choose to control the sending of mail to the mail transport by implementing its own mail spooler. To do so, it sends the **RopSetSpooler** request after logging on to the server using **RopLogon**. The client also needs to save the FID of the spooler queue folder retrieved from a **RopLogon** request for later use.

3.1.4 Higher-Layer Triggered Events

None.

3.1.4.1 Sending a Message

A messaging client sends a message by sending **RopSubmitMessage** to the server. The client can specify submit flags for sending the message (see section about **RopSubmitMessage**). The client can also set the sender information of the message to instruct the server to properly process the message.

3.1.4.1.1 Represented Sender Properties

The represented sender properties SHOULD be set by a client to represent the sender the message is intended to be sent from.

3.1.4.1.2 Actual Sender Properties

Actual sender properties MUST be set to represent the sending mailbox owner.

3.1.4.1.3 Sending the Message as the Sender Itself

When a user intends to represent itself as the actual sender of a message, if the represented sender properties are present, they MUST be set to the values representing the user itself.

3.1.4.1.4 Sending the Message on Behalf of Another Person

If a user sends the message on behalf of another user, the represented sender properties MUST be set to the user the actual sender intends to present. See [MS-OXOCAL].

3.1.4.2 Deferring Sending a Message

PidTagDeferredSendTime MAY be set by a client.

If both PidTagDeferredSendNumber and PidTagDeferredSendUnits are present, PidTagDeferredSendTime SHOULD be computed from PidTagDeferredSendNumber and PidTagDeferredSendUnits.

3.1.4.3 Sending a Message with Expiry Time

PidTagExpiryTime MAY be set by a client.

If both PidTagExpiryNumber and PidTagExpiryUnits are present, PidTagExpiryTime SHOULD be computed from PidTagExpiryNumber and PidTagExpiryUnits.

3.1.4.4 Optimizing Send

When a messaging client sends a message in a client implementation of an optimization, the client can set PidTagTargetEntryId to the value equal to the value of PidTagEntryId of the message being submitted. If this is done, the client MUST move the sent message to its local SentItems folder after submission. Eventually, when the client imports its local Sent Mail folder changes to server, on the server side, the server can make use of PidTagTargetEntryId

to optimize the operation by moving a copy of the submitted message object to the Sent Items folder instead of requiring the client to upload the message object content again. See section 3.2.5.1.2.8 for detailed server operation.

3.1.4.5 **Resending a Message**

If a message fails to be delivered to all recipients, a client MAY mark this message as re-send by setting mfResend in the PidTagMessageFlags property.

The server will attempt to re-deliver this message only to the recipients who did not get the message in the previous delivery attempt.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 **Client-to-Client Interop: Voting**

Voting is built using a specific set of properties on a message to communicate voting options and responses to one another. An overview of the sequence of events is as follows:

- A client (*Sender*) sends a voting message to a variety of recipients (*Voters*). This message contains a well-formed **PidLidVerbStream** as described in section 2.2.1.60, but is otherwise identical to a non-voting message.
- The Voters, upon receiving the message and displaying it to the user, take note of the existence of **PidLidVerbStream** and use the information contained within to display additional voting UI to the user.
- If and when a Voter selects a voting option, a specifically crafted response mail is generated and addressed to the Sender.
- The Sender, upon recieving response messages, aggregates them for display to the user.

It is important to note that at each point in this process, the messages sent are identical to non-voting messages except for the presence of PidLidVerbStream and PidLidVerbResponse.

3.1.5.1.1 Sending a Voting Message

A client wishing to associate a series of voting options with a message MUST set **PidLidVerbStream** as described in section 2.2.1.60.

3.1.5.1.2 Interpreting a Voting Message

When a client receives a message, it MUST check the **PidLidVerbStream** property. If the client encounters a VoteOption structure that does not have 0x00000004 as the VerbType field, the client MUST ignore the existence of that VotingOption <5>.

3.1.5.1.3 Crafting a Voting Response Message

A voting response message MUST contain all of the following:

- **PidTagSubjectPrefix** set to the DisplayName of the voting option choosen by the user
- **PidLidVerbResponse** set to the voting option choosen by the user (see section 2.2.1.61)

Otherwise, the message MUST be formatted as a regular reply e-mail addressed to the initial voting sender; respecting all user preferences that are applicable to such.

The client MUST honor the SendBehavior field of the VoteOption structure. If the SendBehavior field specifies SendPrompt, and if the user selects "Edit", the user MUST be displayed with the appropriate UI to edit the automatically generated response.

3.1.5.1.4 Aggregating Voting Responses

The exact method for aggregating and displaying voting responses is a client implementation detail <6>.

3.1.5.2 Controlling the Sending of Mail:

When submitting a message for delivery that a client wishes to control the specific server that submits the message, it MUST be sent using the **RopSubmitMessage** request with the NeedsSpooler flag (0x02) set. The message is then put into the spooler queue folder of the message store on the server.

3.1.5.3 Processing a Mail in the Spooler Queue

When the client finds an e-mail object in the spooler queue folder that the client can handle<7>, it takes control of the message by sending the **RopSpoolerLockMessage** request with the LockState field set to lstLock. The client then does any implementation dependent processing that it needs. If it decides that the message can be handled by a particular server, it sends the **RopGetTransportFolder** request to retrieve the Folder ID of a folder where temporary transport objects can be stored (clients can cache the returned FID and avoid having to send the request multiple times), creates the message to be sent to the folder, and then sends the **RopTransportSend** request to have that server deliver the message. If the client handles delivering the mail itself, it MUST set the R flag (0x8000) of the RecipientFlags field of each recipient in the recipient table that it successfully delivers mail to.

When it is done, the client sends the **RopSpoolerLockMessage** request with the LockState field set to **lstFinished** if the all recipients have been sent the message, or lstUnlock if some recipients have not yet been sent the message. If some recipients have yet to be processed, the client MUST determine if there is another server that can deliver the e-mail. If another server is found, the client attempts to resubmit the message to the remaining recipients. If no remaining transports can deliver the mail, the client SHOULD generate a non-delivery report (NDR), or notify the user of the error.

3.1.5.4 Delivering Mail to the Server

When a mail is delivered to an account on the server by the client, such as mail received from a POP3 server that is set to deliver into a folder on the server, then it SHOULD send a **RopTransportNewMail** request for each mail delivered to inform the server of the new mail so that the server can do new mail processing.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

The server role for The E-mail Object Protocol follows the Abstract Data Model specified by the Message and Attachment Object Protocol (see [MS-OXCMSG]).

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Handling a RopSubmitMessage request

The server performs the following operations on receipt of the **RopSubmitMessage** request.

3.2.5.1.1 Permission Check

There are restrictions on messages that can be submitted. The server checks for messages submitted and return the corresponding error code if the condition is met.

Conditions	Error code	Value
FAI message is submitted	ecAccessDenied	0x80000009
Embedded message is submitted	ecNotSupported	0x80040102
Upper limit of recipients is exceeded	ecTooManyRecips	0x00000505
Mailbox is running out of quota	ecQuotaExceeded	0x000004D9
No write permission on the message	ecAccessDenied	0x80070005

Further, the server MUST check that the sender has sufficient permissions to send this message on behalf of the actual sender the current sender intends to present.

If the message is sent by another user or user agent, the represented sender properties MUST be set to the user that the actual sender intends to display on the message.

3.2.5.1.2 Properties Read and/or Set Upon Submission

The following properties are checked and modified by the server on the submitted message.

3.2.5.1.2.1 PidTagSentMailSvrEID

If PidTagSentMailSvrEID is present, the message MUST be copied to the folder identified by this property after the message is sent out.

3.2.5.1.2.2 PidTagDeleteAfterSubmit

If PidTagDeleteAfterSubmit is set to 0x01, the message MUST be deleted after the message is sent.

3.2.5.1.2.3 PidTagClientSubmitTime

PidTagClientSubmitTime MUST be set to the current time in UTC.

3.2.5.1.2.4 PidTagContentFilterSpamConfidenceLeve

The server sets PidTagContentFilterSpamConfidenceLeve to 0xFFFFFFFF (-1). A client MAY use this value as part of Junk E-mail or "spam" filtering. See [MS-OXCSPAM].

3.2.5.1.2.5 PidTagMessageLocaleId

The server sets PidTagMessageLocaleId to the current user logon's locale Id.

3.2.5.1.2.6 PidTagMessageFlags

If PidTagMessageFlags's mfResend is set, the message is considered a re-send message and the server will only try to re-deliver the message to those recipients who failed to receive it previously. See section 3.2.4.1.2.6.

3.2.5.1.2.7 PidTagRecipientType

If a message is a re-send message, and if a recipients's PidTagRecipientType has 0x80000000 bit set, the server will ignore this recipient; if a recipient's PidTagRecipientType has 0x10000000 bit set, the server will try to re-deliver the message to this recipient. See [MS-OXCDATA] Recipient Row.

3.2.5.1.2.8 PidTagTargetEntryId<8>

When working in optimizing send mode and sending a message, a client creates a copy of the message in a server folder and MAY set the new message's PidTagTargetEntryId value equal to the value of PidTagEntryId on the original message. Upon the invocation of RopSubmitMessage, the server creates a copy of the submitted message and sets the value of PidTagEntryId to the value obtained from PidTagTargetEntryId.

If the client sets PidTagTargetEntryId, the client MUST keep a copy of the submitted message in the sent mail folder after submission. Eventually, the client will import the move in its local sent mail folder to the server. The server will find the matching item due to the value of PidTagEntryId already existing on server. Instead of requiring the client to upload the message content again, the server completes the operation by moving the copy of the submitted message already persisted on server to the sent mail folder (server side). See [MS-OXCFXICS] for other related information.

Other properties that SHOULD be set at the same time are:

Property	Value
PidTagEntryId	Same value as PidTagTargetEntryId if PidTagTargetEntryId is present. Otherwise, a new ID is generated by the server.
PidTagMessageFlags	mfUnsent and mfRead bits MUST be cleared.
PidTagInternetMessageId	Copied from the original message.

3.2.5.1.2.9 Represented Sender Properties

If the user or user agent who is sending the message is the mailbox owner and the represented sender properties are currently not present, the following represented sender properties MUST be set to the mailbox owner:

- PidTagSentRepresentingAddressType
- PidTagSentRepresentingEmailAddress
- PidTagSentRepresentingEntryId,
- PidTagSentRepresentingName
- PidTagSentRepresentingSearchKey

3.2.5.1.2.10 Actual Sender Properties

If the message is sent with send-on-behalf-of and the represented sender properties represent a public folder or a distribution list, the actual sender properties MUST not be set.Otherwise, the following actual sender properties MUST be set the values of the mailbox owner:

- PidTagSenderAddressType
- PidTagSenderEmailAddress
- PidTagSenderEntryId
- PidTagSenderName
- PidTagSenderSearchKey

3.2.5.1.2.11 Deferred Properties

When a message arrives with the deferred send properties set, then the server MUST honor the deferred send time.

For a message with both PidTagDeferredSendNumber and PidTagDeferredSendUnits are present, during message submission, the server will re-compute PidTagDeferredSendTime from PidTagDeferredSendNumber and PidTagDeferredSendUnits.

3.2.5.1.2.12 Expiry Properties

When a message arrives with the expiry properties set, then the server MUST honor the expiry time.

For a message with both PidTagExpiryNumber and PidTagExpiryUnits are present, during message submission, the server will recompute PidTagExpiryTime from PidTagExpiryNumber and PidTagExpiryUnits.

3.2.5.1.3 Rule Processing

When a message is submitted or delivered, it is subject to further processing by Rules (see [MS-OXORULE]).

3.2.5.2 Handling a RopAbortSubmit Request

When a message is submitted and is still queued on the server pending delivery, the submission can be terminated by sending **RopAbortSubmit**.

If a submitted message's PidTagMessageFlags's mfSubmitted bit has not been set yet, sending **RopAbortSubmit** requests that the server stop delivering the message by removing the message from the spooler queue. The mfUnsent bit of the message's PidTagMessageFlags MUST be set and the mfSubmitted bit of the message's PidTagMessageFlags MUST be cleared. Even if the message's PidTagDeferredSendTime has been set, the client will not be notified of defer send event.

RopAbortSubmit MAY fail at the server's discretion. When RopAbortSubmit fails, the message MAY still be sent.

3.2.5.3 Handling a RopSetSpooler Request

When the RopSetSpooler request is received, the server marks the user logon to indicate that this is a spooler logon.

3.2.5.4 Handling a RopGetTransportFolder Request

The server MUST return a FID that identifies a folder that the client can use to temporarily store messages to be sent.

3.2.5.5 Handling a RopSpoolerLockMessage Request

On receipt of a RopSpoolerLockMessage, a server MUST take the actions based on the value of LockState:

Value	Action
lstLock	Locks the message for the client sending the request. The request MUST fail if the message is locked by some other client.
lstUnlock	Unlock the message
lstFinish	Unlock the message and complete post-processing of sent mail as described in section (insert section reference to RopSpoolerLockMessage)

3.2.5.6 Delivering mail on a RopSubmitMessage or RopTransportSend Request

When a client sends either the RopSubmitMessage request with the NeedsSpooler flag (0x02) not set, or the RopTransportSend request, the server is to attempt to send the e-mail to the intended recipients. For each recipient in the **recipient table** that it can send the e-mail to, it MUST set the R flag (0x8000) of the RecipientFlags field.

When the NeedsSpooler flag is set, the server MUST place the message into the spooler queue folder.

3.2.5.7 Handling a RopTransportNewMail Request

When a server receives this request, it MUST notify all clients connected to the mailbox (using **RopNotify** and a NewMailNotification as described in [MS-OXCDATA]) of the receipt of new mail.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

This section includes examples of message object operations using sequences of ROP requests and responses that a client and a server might exchange. Note that the examples listed here only show the relevant portions of the specified ROPs; this is not the final byte sequence which gets transmitted over the wire. Also note that the data for a multi-byte field appear in little-endian format, with the bytes in the field presented from least significant to most significant. Generally speaking, these ROP requests are packed with other ROP requests, compressed and packed in one or more RPC calls according to the specification in the Remote Operations (ROP) List and Encoding Protocol (for more details, see [MS-OXCROPS]). These examples assume the client has already successfully logged on to the server and have appropriate permissions on the message objects the operations are performed on.

4.1 Submitting a Message

In this example, suppose the messaging client has created a new message object in the mailbox and wishes to submit the the message object. The messaging client previously has set a few message properties to some values which are not particularly interesting in this example and are not documented here.

4.1.1 ROP Request Buffer

The ROP request buffer in this example would look like:

```
0000: 32 00 02 00
```

The composition of the bytes is following:

ROPId: 0x32 (**RopSubmitMessage**)

LogonIndex: 0x00

HandleIndex: 0x02

SubmitFlags: 0x00 (**None**)

The first 3 bytes refer to the ROPId, LogonIndex, HandleIndex, which are the same for all ROPs specified in [MS-OXCROPS]. The SubmitMessageFlags is None. The message identified by its handle Index 0x2 was submitted.

4.1.2 ROP Response Buffer

The ROP response buffer in this example would look like:

```
0000: 32 02 00 00 00 00
```

The composition of the response buffer is as follows:

ROPId: 0x32 (**RopSubmitMessage**)

HandleIndex: 0x02

ReturnValue: 0x00000000 (**ecNone**)

The response's HandleIndex is same as the HandleIndex of the **RopSubmitMessage** and the return value is 0x00000000 which is success. From the response, the message was submitted successfully.

4.2 Submitting a Deferred Message

In this example, suppose the messaging client has created a new message object in the mailbox and wishes to submit the the message object. The client sets properties related to a deferred send. The client also sets other message properties that are not described in Section 4.2.1, but which are not particularly relevant to this example and are not included.

4.2.1 ROP Request Buffer

The ROP request buffer in this example would look like:

```
0000: 0A 01 01 0E 00 01 00 40 00 EF 3F 96 3F 7F F4 5E
0010: 6F C8 01
...
00xx: 32 01 01 00
```

The composition of the bytes is as follows:

ROPId: 0x0A (**RopSetProperties**)

LogonIndex: 0x01

HandleIndex: 0x01

Size: 0x000E

PropertyCount: 0x0001

PropertyTag: 0x3FEF0040 (**PidTagDeferredSendTime**)

PropertyValue: 0x01C86F5EF47F3F96 (UTC FILETIME: 11:11:39PM 02/14/2008)

…

ROPId: 0x32 (**RopSubmitMessage**)

LogonIndex: 0x01

HandleIndex: 0x01

SubmitMessageFlags: 0x00 (**None**)

PidTagDeferredSendTime of the message identified by its handleIndex 0x1 was set to 11:11:39PM 02/14/2008 (UTC). The client intends to defer the submission until 11:11:39PM 02/14/2008.

4.2.2 ROP Response Buffer

The ROP response buffer in this example would look like:

```
0000: 0A 01 00 00 00 00 00 00
...
0000: 32 01 00 00 00 00
```

The composition of the response buffer is as follows:

ROPId: 0x0A (**RopSetProperties**)

HandleIndex: 0x01

ReturnValue: 0x00000000 (**ecNone**)

ProblemPropertyTagCount: 0x0000

ROPId: 0x32 (**RopSubmitMessage**)

HandleIndex: 0x01

ReturnValue: 0x00000000 (**ecNone**)

The response messages to both **RopSetProperties** and **RopSubmitMessage** indicate that the two remote procedure operations succeeded.

If the **RopSubmitMessage** was issued before UTC time 11:11:39PM 02/14/2008, the message would be submitted immediately. If the **RopSubmitMessage** was issued after this time, the message is deferred for submission until the current time is equal to or is later than the deferred send time.

4.3 Aborting a Message Submission

In this example, suppose a client has submitted a message object. While the message is still queued in thte server, the client would like to terminate the submission.

4.3.1 ROP Request Buffer

The ROP request buffer in this example would look like:

```
0000: 34 00 00 01 00 00 03 b4-79 ca 47 01 00 00 03 b7 4
0010: e6 5f a7
```

The composition of the request buffer is as follows:

ROPId: 0x34 (**RopAbortSubmit**)

LogonIndex: 0x00

HandleIndex: 0x00

FolderId: 0001-0003b479ca47 (the FolderId of the parent folder)

MessageId: 0001-0003b7e65fa7 (the message Id of the message submitted)

The message identified by its handleIndex 0x00 was submitted previously. While the message is still queued in the server, the client sends RopAbortSubmit request related to this message to terminate the submission.

4.3.2 ROP Response Buffer

The ROP response buffer in this example would look like:

```
0000: 34 00 00 00 00 00
```

The composition of the response buffer is as follows:

ROPId: 0x34 (**RopAbortSubmit**)

HandleIndex: 0x00

ReturnValue: 0x00000000 (**ecNone**)

The response message indicates RopAbortSubmit succeeded. The message has been removed from the server. The mfUnsent bit is set (restored) and mfSubmitted bit is cleared on the message's PidTagMessageFlags. Unless another RopSubmitMessage is issued on this message object, the message will not be sent.

4.4 Sending an E-mail from a Messaging User to Another Messaging User

Consider the following scenario: Joe Healy needs to send a high importance e-mail to inform his customer, Ed Banti, that the order request form that Ed sent needs to be signed. Joe also wants to get a read receipt when Ed read this e-mail. The following is a description of what a client might do to accomplish Joe's intentions and the responses a server might return.

To create an e-mail object, the client uses **RopCreateMessage**. The server returns a success code and a handle to a message object. Joe types in the e-mail subject and message text (plain text format). The client sets the e-mail to high importance following Joe's wish and also his request to get read receipt, and then uses **RopSetProperties** to transmit Joe's e-mail message data to the server.

Property	Property ID	Type	Value
PidTagBody	0x1000	0x001f (PtypString)	"Please sign the order request.\LF\CR"

Property	Property ID	Type	Value
PidTagMessageClass	0x001A	0x001F (PtypString)	"IPM.Note"
PidTagMessageFlags	0x0E07	0x0003 (PtypInteger32)	mfUnsent
PidTagConversationTopic	0x0070	0x001f (PtypString)	"Order Request"
PidTagConversationIndex	0x0071	0x0102 (PtypBinary)	22 bytes 01 c8 74 0b 0f 9c 35 2c 02 17 93 af 43 a9 8b b4 c1 bb ef 97 7d 4f
PidTagImportance	0x0017	0x0003 (PtypInteger32)	0x00000002 High Importance
PidTagMessageDeliveryTime	0x0E06	0x0040 (PtypTime)	2008/02/20 21:53:00.000
PidTagReadReceiptRequested	0x0029	0x000B (PtypBoolean)	0x01 (TRUE)
PidTagSentMailSvrEID	0x6740	0x00FB (PtypUnspecified)	21 bytes 01 01 00 00 00 00 f0 e7 c1 00 00 00 00 00 00 00 00 00 00 00 00
PidTagIconIndex	0x1080	0x0003 (PtypInteger32)	0xFFFFFFFF
PidTagMessageEditorFormat	0x5909	0x0003 (PtypInteger32)	0x00000001 Plain Text
PidTagPrimarySendAccount	0x0E28	0x001F (PtypString)	000000023659R9-A11/o=First Organization/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=JoeHealy Microsoft

Property	Property ID	Type	Value
			Exchange
PidTagNextSendAcct	0x0E29	0x001F (PtypString)	000000023659R9-A11/o=First Organization/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=JoeHealy Microsoft Exchange
PidTagMessageLocaleId	0x3FF1	0x0003 (PtypInteger32)	1033 (en-us)
PidTagReportTag	0x0031	0x0102 (PtypBinary)	100 bytes (See Note 1 below)

Note 1 - PigTagReportTag binary blob:

```
0000: 50 43 44 46 45 42 30 39-00 01 00 02 00 00 00 00
0010: 00 00 00 00 00 00 00 00-00 2e 00 00 00 00 00 00
0020: 00 1a f8 62 55 f6 35 01-4f b0 20 ce 17 75 e8 64
0030: 0b 01 00 61 2a 7b ab 49-f6 4e 4b 9c 52 db fb 5a
0040: 53 aa 1c 00 00 00 f0 e7-c1 00 00 10 00 00 00 fd
0050: 02 6f a5 55 15 2a 41 ab-1f 64 5d 1b da 0c 38 01
0060: 00 00 00 00
```

Joe then addresses this e-mail to Ed Banti as the primary recipient. The client locates Ed Banti's address data entry from the client's address book and adds Ed Banti's address data to this e-mail message object's recipient table using **RopAddRecipients**.

Recipient Row Element	Value	Description
RowID	0x00000001	Row id number

RecipientType	0x00000001	primary recipient
DataSize	399	
RecipientFlag	0x0651	AddressType.EXCH DisplayName XmitSameAsDisplay StandardPropsUnicode SimpleDisplayName
DNPrefixLen	0x5A (90)	
EX-Address.Type	0x00000000	DT_MAILUSER
EX-Address.EmailAddress	edbanti	
DisplayName	Ed Banti	
SimpleDisplayName	Ed Banti	

The client adds additional properties in the recipient row:

Property	**PropertyID**	**Type**	**Value**
PidTagObjectType	0x0FFE	0x0003 (PtypInteger32)	0x00000006 (MAILUSER)
PidTagDisplayType	0x3900	0x0003 (PtypInteger32)	0x00000000 DT_MAILUSER
PidTag7BitDisplayName	0x39FF	0x001F (PtypString)	Ed Banti
PidTagSmtpAddress	0x39FE	0x001F (PtypString)	edbanti@example.com

PidTagSendInternetEncoding	0x3A71	0x0003 (PtypInteger32)	0x00000000
PidTagNickName	0x6001	0x001F (PtypString)	edbanti@example.com
PidTagAccount	0x3A00	0x001F (PtypString)	edbanti
PidTagDisplayTypeEx	0x3905	0x0003 (PtypInteger32)	1073741824
PidTagRecipientTrackStatus	0x5FFF	0x0003 (PtypInteger32)	0x00000000
PidTagRecipientResourceState	0x5FDE	0x0003 (PtypInteger32)	0x00000000
PidTagRecipientFlags	0x5FFD	0x0003 (PtypInteger32)	0x00000001
PidTagRecipientDisplayName	0x5FF6	0x001F (PtypString)	Ed Banti
PidTagRecipientEntryId	0x5FF7	0x0102 (PtypBinary)	126 bytes (see the sample value for PidTagRecipientEntryId following this table)
PidTagRecipientOrder	0x5FDF	0x0003 (PtypInteger32)	0x00000000

PidTagRecipientEntryId:

```
0000: 00 00 00 00 dc a7 40 c8-c0 42 10 1a b4 b9 08 00
0010: 2b 2f e1 82 01 00 00 00-00 00 00 00 2f 6f 3d 46
0020: 69 72 73 74 20 4f 72 67-61 6e 69 7a 61 74 69 6f
```

```
0030: 6e 2f 6f 75 3d 45 78 63-68 61 6e 67 65 20 41 64
0040: 6d 69 6e 69 73 74 72 61-74 69 76 65 20 47 72 6f
0050: 75 70 20 28 46 59 44 49-42 4f 48 46 32 33 53 50
0060: 44 4c 54 29 2f 63 6e 3d-52 65 63 69 70 69 65 6e
0070: 74 73 2f 63 6e 3d 65 64-62 61 6e 74 69 00
```

Last, Joe sends the e-mail. The client sets the calculated subject properties on the e-mail message object based on the subject text on Joe's submitted message using **RopSetProperties**.

Property	PropertyID	Type	Value
PidTagSubjectPrefix	0x0003	0x001F (PtypString)	Empty string
PidTagNormalizedSubject	0x0E1D	0x001F (PtypString)	"Order Form Issue"

The client then sends a **RopSubmitMessage** request to ask server to deliver this e-mail message to Ed Banti and sends a **RopRelease** request to release the e-mail message object.

Please see all the Rop operation details used in this example as specified in [MS-OXCROP], [MS-OXCMSG], and [MS-OXOMSG] documents. For client's offline e-mail address book and recipient address data entry details, please see [MS-OXOAB] and [MS-OXOABK] documents.

4.5 Message with Voting Options

In this example, a user wishes to send a message with a "Yes", "No", and "Maybe" voting options. To do so, the client MUST construct a message containing a **PidLidVerbStream** as described in section 2.2.1.60.

The complete contents of **PidLidVerbStream** in this example are show below. The other properties of the message are not specific to voting, and are omitted.

```
0000: 02 01 03 00 00 00 04 00-00 00 03 59 65 73 08 49
0010: 50 4D 2E 4E 6F 74 65 00-03 59 65 73 00 00 00 00
0020: 00 00 00 00 00 01 00 00-00 02 00 00 00 02 00 00
0030: 00 01 00 00 00 FF FF FF-FF 04 00 00 00 02 4E 6F
0040: 08 49 50 4D 2E 4E 6F 74-65 00 02 4E 6F 00 00 00
0050: 00 00 00 00 00 00 01 00-00 00 02 00 00 00 02 00
0060: 00 00 02 00 00 00 FF FF-FF FF 04 00 00 00 05 4D
0070: 61 79 62 65 08 49 50 4D-2E 4E 6F 74 65 00 05 4D
0080: 61 79 62 65 00 00 00 00-00 00 00 00 00 01 00 00
```

0090: 00 02 00 00 00 02 00 00-00 03 00 00 00 FF FF FF

00A0: FF 04 01 03 59 00 65 00-73 00 03 59 00 65 00 73

00B0: 00 02 4E 00 6F 00 02 4E-00 6F 00 05 4D 00 61 00

00C0: 79 00 62 00 65 00 05 4D-00 61 00 79 00 62 00 65

00D0: 00

The first 6 bytes contain the *Version* and *Count* fields as described in section 2.2.1.60.

0000: 02 01 03 00 00 00

Version: 0x0102

Count: 0x00000003

This indicates that this structure contains three *VoteOptions*. The first *VoteOption* begins at byte **0x0006**.

0006: 04 00 00 00 03 59 65 73-08 49 50 4D 2E 4E 6F 74

0016: 65 00 03 59 65 73 00 00-00 00 00 00 00 00 00 01

0026: 00 00 00 02 00 00 00 02-00 00 00 01 00 00 00 FF

0036: FF FF FF

VerbType: 0x00000004

DisplayNameCount: 0x03

DisplayName: ANSI String (not null terminated): "Yes"

MsgClsNameCount: 0x08

MsgClsName: ANSI String (not null terminated): "IPM.Note"

Internal1: 0x00

DisplayNameCountRepeat: 0x03

DisplayNameRepeat: ANSI String (not null terminated): "Yes"

Internal2: 0x00000000

Internal3: 0x00

fUseUSHeaders: False (0x00000000)

Internal4: 0x00000001

SendBehavior: 0x00000002 (SendPrompt)

Internal5: 0x00000002

ID: 0x00000001

Internal6: 0xFFFFFFFF

The second and third *VoteOption* structures (for "No" and "Maybe") begin at bytes **0x0039** and **0x006A** respectively. The third *VoteOption* concludes at byte **0x00A0**, and byte **0x00A1** begins the Version2 field.

00A1: 04 01

Version2: 0x0104

This is followed by three *VoteOptionExtras* structures; a parallel array containing additional information about the three *VoteOption* structures seen earlier. The first begins at byte **0x00A3**.

00A3: 03 59 00 65 00 73 00 03-59 00 65 00 73 00

DisplayNameCount: 0x03

DisplayName: Unicode String (not null terminated): "Yes"

DisplayNameCountRepeat: 0x03

DisplayNameRepeat: Unicode String (not null terminated): "Yes"

The second and third *VoteOptionExtras* structures (for "No" and "Maybe") begin at bytes **0x00B1** and **0x00BB** respectively, and constitude the remainder of the buffer.

4.6 Controlling Sending Mail to a Specific Server

Ellen Adams is using a mail client that is connected to both her work and personal e-mail accounts. Her personal e-mail account is accessed through a protocol which is not the Office/Exchange protocol, but through some other standard such as POP3. Her personal e-mail is set to deliver to a folder in her work account.

4.6.1 Initialization

When the mail client first initializes, it sends a RopSetSpooler request to inform that the mail client wants to be responsible for routing mail to the messaging transport:

4.6.1.1 ROP Request Buffer

The ROP request buffer in this example would look like:

0000: 47 06 00

The composition of the bytes is following:

RopId: 0x47 (RopSetSpooler)

LogonId: 0x06

InputHandleIndex: 0x00 (Handle to the logon object)

4.6.1.2 **ROP Response Buffer**

The server then returns a response buffer:

```
0000: 47 00 00 00 00 00
```

The composition of the response buffer is as follows:

RopId: 0x47 (RopSetSpooler)

InputHandleIndex: 0x00

ReturnValue: ecNone (Success)

4.6.2 Submission of the Message to the Spooler Queue Folder

Ellen then sends a mail from her work account. The client follows the example in section 4.1, except setting the NeedsSpooler (0x2) bit in the SubmitFlags field, as well as setting a property or somehow informing the spooler which mail transport to use<9>. The server places the message in the spooler queue folder (the Folder ID of this folder is returned in the response buffer of a RopLogon request)

4.6.3 Locking the Message in the Spooler Queue Folder for Processing

Next, the client finds that a message has been placed in the spooler queue folder. Through an implementation dependent mechanism, it determines that it can handle the message<10>. It sends the RopSpoolerLockMessage request to lock the message.

4.6.3.1 **ROP Request Buffer**

```
0000: 48 06 00 01 00 00 03 BB-97 31 A7 00
```

The composition of the bytes is following:

RopId: 0x48 (RopSpoolerLockMessage)

LogonId: 0x06

InputHandleIndex: 0 (Handle to the logon object)

MessageId: 0001-0003bb9731a7

LockState: 0x00 (lock)

4.6.3.2 **ROP Response Buffer**

The server then returns a response buffer:

```
0000: 48 00 00 00 00 00
```

The composition of the response buffer is as follows:

RopId: 0x48 (RopSpoolerLockMessage)

InputHandleIndex: 0x00

ReturnValue: ecNone (success) (0x00000000)

4.6.4 Determination of the Transport Folder

The client determines the server (Ellen's work server) that the message should be routed to (which may be the same or different than the server holding the spooler queue). The client sends a **RopGetTransportFolder** request to request the location of a temporary folder for transport.

4.6.4.1 **ROP Request Buffer**

```
0000: 6D 07 01
```

The composition of the bytes is following:

RopId: 0x6D (RopGetTransportFolder)

LogonId: 0x07

InputHandleIndex: 0x01 (Handle to the logon object)

4.6.4.2 **ROP Response Buffer**

The server then returns a response buffer with the FID of a folder that can be used to store temporary transport objects:

```
0000: 6D 01 00 00 00 00 01 00-00 00 00 00 00 25
```

The composition of the response buffer is as follows:

RopId: 0x6D (RopGetTransportFolder)

InputHandleIndex: 0x01

ReturnValue: ecNone (success) (0x00000000)

FID: 0001-000000000025

4.6.5 Sending the Message

The client examines the locked message, does any processing needed (for example, determining whether there are any recipients that it knows the server cannot deliver to), and creates a copy of the message to be delivered in the folder just retrieved using the **RopCreateMessage** request (details on how to use this ROP are in [MS-OXCMSG]).

The client then sends a RopTransportSend request to have the server actually send the message.

4.6.5.1 **ROP Request Buffer**

```
0000: 4A 07 00
```

The composition of the bytes is following:

RopId: 0x4A (RopTransportSend)

LogonId: 0x07

InputHandleIndex: 0x00 (Handle to the message from RopCreateMessage)

4.6.5.2 **ROP Response Buffer**

The server then returns a response buffer:

```
0000: 4A 00 00 00 00 00 01
```

The composition of the response buffer is as follows:

RopId: 0x4A (RopTransportSend)

InputHandleIndex: 0x00

ReturnValue: ecNone (success) (0x00000000)

NoPropertiesReturned: 0x01 (TRUE)

4.6.6 Marking the Message as Ready for Post-Send Server Processing

Finally, the client sends the RopSpoolerLockMessage request with the finish flag to the server to have it do any post-processing on the sent message:

4.6.6.1 **ROP Request Buffer**

```
0000: 48 06 00 01 00 00 03 BB-97 31 A7 02
```

The composition of the bytes is following:

RopId: 0x48 (RopSpoolerLockMessage)

LogonId: 0x06

InputHandleIndex: 0x00 (Handle to the logon object)

MessageId: 0001-0003bb9731a7

LockState: 0x02 (finish)

4.6.6.2 **ROP Response Buffer**

The server then returns a response buffer:

```
0000: 48 00 00 00 00 00
```

The composition of the response buffer is as follows:

RopId: 0x48 (RopSpoolerLockMessage)

InputHandleIndex: 0x00

ReturnValue: ecNone (success) (0x00000000)

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to the [MS-OXOMSG] protocol. General security considerations pertaining to the underlying RPC-based transport apply (see [MS-OXCROPS]).

5.2 Index of Security Parameters

None.

6 Appendix A: Office/Exchange Behavior

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

- Office 2003 with Service Pack 3 applied
- Exchange 2003 with Service Pack 2 applied
- Office 2007 with Service Pack 1 applied
- Exchange 2007 with Service Pack 1 applied

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

<1> Section 2.2.1.4: PidTagDeferredDeliveryTime is honored by Exchange Server 2003 SP2. Starting from Exchange 2007 SP1, only PidTagDeferredSendTime is used. A client SHOULD set both PidTagDeferredDeliveryTime and PidTagDeferredSendTime for deferred delivery message before submission.

<2> Section 2.2.1.8: Outlook computes this value most of the time, and does not store the result on the object.

<3> Section 2.2.2.23: On report messages, the value of the PidTagSubjectPrefix property is typically set as :

Delivery receipts: *"Delivered: "*

Read receipts: *"Read: "*

Sender Response on read receipt request: *"Approved: "*

Non-deliverable receipts: *"Undeliverable: "*

Non-read receipts: *"Not Read: "read: "*

<4> Section 2.2.4.3.1: Examples of address types are: "EX", "MAPIPDL", "SMTP", "MHS", "PROFS", "X400". It's possible there are other custom address types generated by third-party applications.

<5> Section 3.1.5.1.2: Office 2007 SP1 also uses **PidLidVerbStream** for non-voting related actions not covered in this protocol. Each of this actions has a specific VerbType associated with it. The format of the VoteOption structure is identical for these non-voting related actions; however, the internal values specific in the struct will vary. Future versions of Office may further define additional VerbTypes; it is therefore advised that clients ignore VoteOption structures which do not specify VerbTypes that they

understand. Likewise, Office 2007 SP1 ignores VoteOption structures with unknown VerbTypes.

<6> Section 3.1.5.1.4: Office 2007 SP1 uses a system similar to meeting responses in order to track voting options. When it receives a voting response, it finds the initial voting message in the sent mail folder. It then updates the recipients table for the recipient who sent the response to store the index of their response. If the user opens a voting message from the sent mail folder, it then sums the total of each response received thus far from the recipient table and displays it to the user.

<7> Section 3.1.5.3: Outlook writes to the property PidTagNextSendAcct to a user-specified value before submitting the message via RopSubmitMessage to inform the spooler the desired mail transport to use.

<8> Section 3.2.5.1.2.7: PidTagTargetEntryId is optimized and supported by Office 2007 SP1 above and Exchange 2007 SP1.

<9> Section 4.6.2: Outlook's mail spooler uses the value of PidTagNextSendAcct to determine the desired mail transport.

<10> Section 4.6.3: Outlook examines the property PidTagNextSendAcct

7 Index