[MS-OXOFLAG]:
Informational Flagging Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

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

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

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

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability.
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Revised and edited technical content.
09/03/2008	1.02		Revised and edited technical content.
12/03/2008	1.03		Revised and edited technical content.
04/10/2009	2.0		Updated technical content and applicable product releases.
07/15/2009	3.0	Major	Revised and edited for technical content.
11/04/2009	4.0.0	Major	Updated and revised the technical content.
02/10/2010	5.0.0	Major	Updated and revised the technical content.

Table of Contents

1   Introduction

Flagging is a way for **Message objects** to be marked for follow up. As part of the Informational Flagging protocol, flagging related **properties** are set on the Message object. The values of these properties can then be used to identify flagged **messages**.

This document specifies:

- Properties that, when set, can be used by a client or server to identify Message objects as being flagged.

- How flag-related properties can be used to track the progress and completion of an associated work item.

- A method by which the sender of a message can specify the flag-related properties of the object delivered to the **recipient(s)** independently of the flag-related properties of the copy of the object saved in the sender's **mailbox**.

1.1   Glossary

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

> **Appointment object**
> **complete flag**
> **Contact objects**
> **Coordinated Universal Time (UTC)**
> **Draft Message object**
> **exceptions**
> **handle**
> **Journal object**
> **mailbox**
> **meeting-related object**
> **message**
> **Message object**
> **property**
> **property identifiers**
> **property name**
> **property tag**
> **property type**
> **recipient**
> **reminder**
> **ROP response buffer**
> **Task object**
> **time flag**

The following terms are specific to this document:

> **basic flag:** A flag on a **Message object** that indicates that the object has an associated work item or shares a defining characteristic with other **Message objects** with such flags.

> **color flag:** A flag that extends the concept of a **basic flag** by associating one of a chosen set of color values with a flagged **Message object**.

> **consolidated to-do list:** A list of all tasks and flagged **Message objects** in a user's mailbox.

Release: Friday, February 5, 2010

primary flag storage location: The usual storage location of flagging **properties** (as opposed to the **secondary flag storage location**).

recipient flag: A collection of **property** values that indicate that a **Draft Message object** has been marked such that it will appear **basic flagged** to recipients.

recipient reminder: A collection of **property** values that indicate that a **Draft Message object** has been marked such that it will have an active reminder for the recipients of the **Message object**.

secondary flag storage location: A binary **property** that is used to encode a second set of flagging **properties**, which do not affect the flagged state of the **Message object**.

sender flag: A collection of **property** values that indicate that a **Draft Message object** has been marked such that the copy of the **Message object** saved in the sender's mailbox after the message is sent will appear flagged to the sender.

sender reminder: A collection of **property** values that indicate that a **Draft Message object** has been marked such that the copy of the **Message object** saved in the sender's mailbox after the message is sent will have an active reminder.

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

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-DTYP] Microsoft Corporation, "Windows Data Types", March 2007, http://go.microsoft.com/fwlink/?LinkId=111558.

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

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

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

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

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

[MS-OXOJRNL] Microsoft Corporation, "Journal Object Protocol Specification", June 2008.

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

[MS-OXONOTE] Microsoft Corporation, "Note Object Protocol Specification", June 2008.

[MS-OXORMDR] Microsoft Corporation, "Reminder Settings Protocol Specification", June 2008.

[MS-OXOTASK] Microsoft Corporation, "Task-Related Objects Protocol Specification", June 2008.

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

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

1.2.2 Informative References

None.

1.3 Protocol Overview

The Informational Flagging protocol specifies a set of properties that a messaging system can use to identify Message objects as not flagged, **basic flagged**, **color flagged**, **time flagged**, or completed. Objects that are flagged can also have additional associated information such as flag color, start date, or due date to help users further organize their flagged objects; for example, users can assign start and due dates to their flagged objects to prioritize their work, or they can assign a flag color to group related objects.

This protocol specifies a method for a **Draft Message object** to be marked with a **recipient flag**, such that recipients will see the Message object as basic flagged upon receipt. This protocol also specifies a method for a Draft Message object to be marked with a **sender flag**, which is a time flag that is displayed on the copy of the Message object saved in the sender's mailbox and is not exposed to the message recipients.

1.4 Relationship to Other Protocols

This specification relies on an understanding of how to work with Message object properties, **Task object** properties, and message submission. For more details, see [MS-OXCMSG], [MS-OXOTASK], and [MS-OXOMSG]. Sender flags are closely related to **sender reminders**, which are specified in [MS-OXORMDR].

1.5 Prerequisites/Preconditions

This specification assumes that the messaging client has previously logged on to the server and has acquired a **handle** to the Message object on which it intends to operate.

1.6 Applicability Statement

This specification can be used to specify flags for the Message objects. This protocol is intended to be a complement, and not a substitute, for full task management, which is specified in [MS-OXOTASK].

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9   Standards Assignments

None.

2 Messages

2.1 Transport

The Informational Flagging protocol extends the protocol specified in [MS-OXCMSG] and uses the protocol specified in [MS-OXCPRPT] as an underlying transport.

2.2 Message Syntax

Unless otherwise noted below, the client includes the properties specified in the documentation for the specific type of Message object that is to have its flag state changed.

When a value is specified as not present, the property MUST NOT exist on the Message object, and it MUST be deleted if it exists. Setting a property value to 0 or a zero-length string does not delete the value of the property from the Message object.

2.2.1 Properties Specific to the Informational Flagging Protocol

2.2.1.1 PidTagFlagStatus

Type: **PtypInteger32**

Specifies the flag state of the Message object. It MUST NOT exist on a **meeting-related object**, and it SHOULD NOT exist on a Task object <1>. When acting on other Message objects, set this property to one of the following values.

Numeric value	Name	Meaning
Not present	N/A	Unflagged
0x00000001	followupComplete	Flagged complete
0x00000002	followupFlagged	Flagged

2.2.1.2 PidTagFollowupIcon

Type: **PtypInteger32**

Specifies the flag color of the Message object. It MUST NOT exist unless the value of the PidTagFlagStatus property is set to "followupFlagged", or the Message object is a meeting-related object. This property SHOULD NOT exist on a Task object <2>. When acting on other Message objects, set this property to one of the following values.

Numeric value	Name	Meaning
Not present	N/A	No color
0x00000001	followupIcon1	Purple flag
0x00000002	followupIcon2	Orange flag
0x00000003	followupIcon3	Green flag
0x00000004	followupIcon4	Yellow flag

Numeric value	Name	Meaning
0x00000005	followupIcon5	Blue flag
0x00000006	followupIcon6	Red flag

2.2.1.3   PidTagFlagCompleteTime

Type: **PtypTime**

Specifies the date and time in **UTC** that the Message object was flagged completed, and the property is deleted if the Message object is not flagged complete. The time's smallest resolution MUST be minutes (the value MUST be a multiple of 600,000,000). This property MUST NOT exist if the object is a meeting-related object, it and SHOULD NOT exist on a Task object <3>.

2.2.1.4   PidTagReplyRequested

Type: **PtypBoolean**

This property SHOULD NOT be changed if the object is a meeting-related object because this property has a specialized meaning for meeting-related objects, as specified in [MS-OXOCAL]. This property SHOULD NOT exist if the object is a Task object <4>This property SHOULD be set to "0x01" if the object is flagged and set to "0x00" or deleted otherwise <5><6>.

2.2.1.5   PidTagResponseRequested

Type: **PtypBoolean**

This property has identical values and semantics to the PidTagReplyRequested property in terms of this protocol and therefore, the client updates these values in an identical manner.

2.2.1.6   PidTagToDoItemFlags

Type: **PtypInteger32**

A bit field in which each bit SHOULD be set to 1 if the associated **condition** in the table below applies and 0 otherwise <7>. The possible bit values are as follows.

Numeric value	Name	Meaning
Not present	N/A	Unflagged
0x00000001	todoTimeFlagged	Object is time flagged <8>.
0x00000008	todoRecipientFlagged	SHOULD only be set on a Draft Message object, and it means that the object is flagged for recipients.

All bits not specified in the above table are reserved. They MUST be ignored, but SHOULD be preserved if they are set.

2.2.1.7   PidTagSwappedToDoData

Type: **PtypBinary**

Acts as the **secondary flag storage location** if sender flags or sender reminders are supported [9]. This secondary storage location can be used by the client to maintain a second set of property values relating to the Informational Flagging protocol that do not affect the flagged state of the Message object.

This structure provides a location at which the client can store all of the properties relating to the Informational Flagging protocol that are supported in sender flags, and all properties relating to the **reminder** Settings protocol that are supported in sender reminders without exposing the sender flag or sender reminder information to the recipients of a message. Similarly, this structure provides a location at which all of the properties relating to the Informational Flagging protocol that are supported in recipient flags and properties relating to the reminder Settings protocol that are supported in **recipient reminder**s can be stored for informational purposes on a previously sent message.

The structure contains the following fields.

0	1	2	3	4	5	6	7	8	9	1 0	1	2	3	4	5	6	7	8	9	2 0	1	2	3	4	5	6	7	8	9	3 0	1
ulVersion																															
dwFlags																															
dwToDoItem																															
wszFlagTo																															
...																															
...																															
...																															
...																															
...																															
...																															
...																															
(wszFlagTo cont'd for 120 rows)																															
rtmStartDate																															

rtmDueDate
rtmReminder
fReminderSet

ulVersion (4 bytes): PtypInteger32; Version of the binary structure. This protocol only specifies version "0x00000001". The contents of this field MUST be ignored if the version number is not "0x00000001".

dwFlags (4 bytes): PtypInteger32; A bit field that determines the validity of the member fields below. The bits are specified in the next table.

dwToDoItem (4 bytes): PtypInteger32; Corresponds to PidTagToDoItemFlags.

wszFlagTo (512 bytes): WCHAR[256]; Corresponds to PidLidFlagRequest.

rtmStartDate (4 bytes): PtypInteger32; Corresponds to PidLidTaskStartDate.

rtmDueDate (4 bytes): PtypInteger32; Corresponds to PidLidTaskDueDate.

rtmReminder (4 bytes): PtypInteger32; Corresponds to PidLidReminderTime, PidLidReminderSignalTime, and PidTagReplyTime. When swapping the contents of the primary and secondary flag storage location, the data of **rtmReminder** is written to three properties: PidLidReminderTime, PidLidReminderSignalTime, and PidTagReplyTime, and the data in PidLidReminderTime is written to rtmReminder. For more details about reminder properties, see [MS-OXORMDR].

fReminderSet (4 bytes): PtypInteger32; Corresponds to PidLidReminderSet [MS-OXORMDR]. This property is treated as a Boolean with 0x00000000 and 0x00000001 as valid values.

The **rtmStartDate**, **rtmDueDate**, and **rtmReminder** fields are stored as a **PtypInteger32** expressed as the number of minutes since 00:00:00 on January 1, 1601 in UTC. Clients write the value 0x5AE980E0 to indicate no date and time.

The **dwFlags** field has the following valid bit values. The bits specified in the table below are set to 1 if the corresponding data member in the structure above has valid data, and are set to 0 otherwise.

Bit	Validity of data field
0x00000001	dwToDoItem
0x00000008	rtmStartDate
0x00000010	rtmDueDate
0x00000020	wszFlagTo
0x00000040	fReminderSet
0x00000080	rtmReminder

2.2.1.8 PidTagSwappedToDoStore

Type: **PtypBinary**

If PidTagSwappedToDoData is set on a Draft Message object, then the value of this property is set to the value of PidTagStoreEntryId of the message. This property is used to determine the need for post-transmit processing of an e-mail, as specified in section 3.1.4.3.

2.2.1.9 PidLidFlagRequest

Type: **PtypString**

Contains user-specifiable text <10> to be associated with the flag and SHOULD be set if the Message object is flagged or completed, but SHOULD NOT exist for a meeting-related object. For more details and a list of defaults suggested to the user, see section 2.2.1.10. This property SHOULD be conditionally ignored based on the values of PidLidFlagString and PidLidValidFlagStringProof. For more details, see sections 2.2.1.10 and 2.2.1.11<11>.

2.2.1.10 PidLidFlagString

Type: **PtypInteger32**

The PidLidFlagString SHOULD NOT exist on meeting-related objects. This string contains an index identifying one of a set of pre-defined text strings to be associated with the flag. If set, clients SHOULD use the corresponding string value in the tables below indicated by PidLidFlagString (for example, to substitute a string that is translated into the current user's language), and SHOULD ignore the value set in PidLidFlagRequest and PidLidValidFlagStringProof <12><13>.

Defaults suggested to the user for **Contact objects** are as follows.

Value	English string
0x00000000 or not present	Follow the guidance related to displaying PidLidFlagRequest.
0x0000006E	"Follow up"
0x0000006F	"Call"
0x00000070	"Arrange Meeting"
0x00000071	"Send E-mail"
0x00000072	"Send Letter"

Defaults suggested to the user for all other Message objects are as follows.

Value	English string
0x00000000 or not present	Follow the guidance related to displaying PidLidFlagRequest.
0x00000001	"Call"
0x00000002	"Do not Forward"
0x00000003	"Follow up"
0x00000004	"For Your Information"

Value	English string
0x00000005	"Forward"
0x00000006	"No Response Necessary"
0x00000007	"Read"
0x00000008	"Reply"
0x00000009	"Reply to All"
0x0000000A	"Review"

All strings specified above can be translated to the user's language, if appropriate.

2.2.1.11   PidLidValidFlagStringProof

Type: **PtypTime**

The PidLidValidFlagStringProof SHOULD NOT exist on meeting-related objects<14>. On non-sendable objects (received mail and non-mail objects), clients SHOULD set this value to the value of PidTagMessageDeliveryTime when modifying PidLidFlagRequest <15>.

This property can be used to validate whether PidLidFlagRequest was set by an agent with knowledge of PidTagMessageDeliveryTime <16>. Since the value of PidTagMessageDeliveryTime cannot be predicted by the sender, if the value of PidLidValidFlagStringProof is equal to the value of PidTagMessageDeliveryTime, it is reasonably certain that the value of PidLidFlagRequest did not originate from the sender of the message. A client can decide how to present the value of PidLidFlagRequest to the end user based on the result of this comparison in accordance with the specific security policy of the client. If the value of PidLidFlagRequest is ignored due to the presence of a value for PidLidFlagString, this property is ignored.

2.2.1.12   PidLidToDoTitle

Type: **PtypString**

Contains user-specifiable text to identify this Message object in a **consolidated to-do list**. PidLidToDoTitle MUST NOT be set on a Task object. To indicate an empty property, PidLidToDoTitle SHOULD NOT be set to the zero-length string, and instead SHOULD be deleted. When flagging a Message object, and the property does not exist, a client SHOULD write the value of PidTagNormalizedSubject to this property.

In a consolidated to-do list, if this property does not exist, a client SHOULD substitute the value of the PidTagNormalizedSubject <17> property when displaying this property in the to-do list.

On a Draft Message object, if the client implements sender flags, this property SHOULD be set to the same value as PidLidFlagRequest.

2.2.1.13   PidLidToDoOrdinalDate

Type: **PtypTime**

This property SHOULD be used to determine the sort order of objects in a consolidated to-do list. When an object is flagged, this property SHOULD <18> be set to the current time in UTC. If the client allows a user to reorder tasks within the consolidated task list via dragging or other

mechanisms, the client can use any suitable algorithm to determine the new value of this property such that the task appears in the correct place when this property is used as a sorting field. When this property is used to perform a sort of objects and the resultant sort results in a tie, the PidLidToDoSubOrdinal property is used to break the tie.

2.2.1.14 PidLidToDoSubOrdinal

Type: **PtypString**

When the PidLidToDoOrdinalDate property is used to perform a sort of objects and the resultant sort results in a tie, this property is used to break the tie. If used, this property MUST be sorted lexicographically. The component characters of the string consists of only the numerals 0 through 9. This property SHOULD be initially set to "5555555". The length of this property MUST NOT exceed 254 characters (excluding the terminating NULL character).

2.2.2 Properties Shared with the Task-Related Object Protocol

The following properties are shared by this specification and Task-Related Object Protocol Specification [MS-OXOTASK]:

- PidLidTaskStartDate

- PidLidTaskDueDate

- PidLidTaskDateCompleted

- PidLidTaskComplete

- PidLidTaskStatus

- PidLidPercentComplete

- PidLidCommonStart

- PidLidCommonEnd

Unless noted below, their semantics and accepted values are identical to those specified in [MS-OXOTASK].

2.2.2.1 PidLidTaskStatus

Type: **PtypInteger32**

For a time flagged Message object, the value of this property is set to "0x00000002" if the task is completed, and SHOULD be set to "0x00000000" otherwise.

2.2.2.2 PidLidPercentComplete

Type: **PtypFloating64**

For a time flagged Message object, the value of this property is set to "1.0" if the object is flagged completed, and SHOULD be set to "0.0" otherwise.

2.2.3 Properties Shared with the Reminder Settings Protocol

The following properties are shared by the Reminder Settings Protocol Specification, as specified in [MS-OXORMDR]:

- PidLidReminderSet

- PidLidReminderTime

- PidLidReminderSignalTime

- PidTagReplyTime

Unless noted below, their semantics and accepted values are identical to those specified in [MS-OXORMDR].

2.2.3.1   PidTagReplyTime

Type: **PtypTime**

On a Draft Message object, if the sender desires to set a deadline for the recipient(s), this property is set to the desired deadline in UTC. This property SHOULD be deleted when clearing the flag from the Message object <19>.

3   Protocol Details

3.1   Client and Server Details

Message objects can be flagged by clients and servers. Except where noted below, this section defines constraints to which clients and servers adhere when flagging Message objects.

Clients operate on Message objects by using the protocol specified in [MS-OXCMSG]. How a server operates on Message objects is implementation dependent, but the results of any such operations MUST be exposed to clients in a manner that is consistent with this specification.

[MS-OXPROPS] specifies the definitions of each property, including the **property name**, **property tag**, and **property type**.

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 specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

A Message object is always considered to be in one of these states with respect to this specification.

- Basic flagged

- Color flagged

- Time flagged

- **Flagged complete**

- Unflagged

Otherwise, the abstract data model of the Informational Flagging protocol does not differ significantly from the abstract data model of the Message object to which this protocol is being applied.

3.1.2   Timers

None.

3.1.3   Initialization

None.

3.1.4   Higher-Layer Triggered Events

3.1.4.1   Flagging a Message Object

Message objects, other than **Appointment objects** [MS-OXOCAL], **Journal objects** [MS-OXOJRNL], and sticky notes [MS-OXONOTE] can be flagged. Appointment objects [MS-OXOCAL], Journal objects [MS-OXOJRNL], and sticky notes [MS-OXONOTE] SHOULD NOT be flagged <20>.

3.1.4.1.1 Setting a Color Flag

The following properties are set to these values on any Message object when flagging a message with a color flag <21>.

Property	Value
PidTagFollowupIcon	Not present for no color or followupIcon1 – followupIcon6.

For all Message objects that are not meeting-related objects, the following properties SHOULD also be set.

Property	Value
PidLidFlagRequest	SHOULD be a non-zero length string. "Follow up" SHOULD be used as the default if one is not specified explicitly by the user.
PidLidFlagString	For more details, see section 2.2.1.10 <22>.
PidLidValidFlagStringProof	Equal to PidTagMessageDeliveryTime of the Message object <23>.

For all Message objects that are not meeting-related objects or Task objects <24>, the following properties SHOULD be set.

Property	Value
PidTagFlagStatus	followupFlagged (0x00000002).
PidTagFlagCompleteTime	Not present.
PidTagReplyRequested	0x01 <25>.
PidTagResponseRequested	0x01 <26>.

3.1.4.1.2 Setting a Basic Flag

A basic flag is identical to the color flag specified in section 3.1.4.1 except that the property PidTagFollowupIcon is not set <27>. Meeting-related objects cannot have a basic flag.

3.1.4.1.3 Setting a Time Flag

The following properties SHOULD be set to these values on any Message object when flagging a message with a time flag <28>.

Property	Value
PidTagFollowupIcon	followupIcon6 (0x00000006)
PidTagToDoItemFlags	todoTimeFlagged (0x00000001) bit set to 1.
PidLidToDoOrdinalDate	Equal to the current time in UTC if PidLidToDoOrdinalDate does not already exist.
PidLidToDoSubOrdinal	"5555555" if PidLidToDoSubOrdinal does not already exist.
PidLidTaskStartDate	User specified.

Property	Value
PidLidTaskDueDate	User specified.
PidLidCommonStart	See [MS-OXOTASK].
PidLidCommonEnd	See [MS-OXOTASK].
PidLidTaskDateCompleted	Not present.
PidLidTaskComplete	0x00
PidLidTaskStatus	0x00000000
PidLidPercentComplete	0.0

Note that even if the user specifies "no date" for the start and due dates, the flag is still considered a time flag if the **todoTimeFlagged** ("0x00000001") bit in the property PidTagToDoItemFlags is set to 1.

For all Message objects that are not Task objects, the following properties SHOULD also be set.

Property	Value
PidLidToDoTitle	Equal to PidTagNormalizedSubject if PidLidToDoTitle does not already exist.

For all Message objects that are not meeting-related objects, the following properties SHOULD also be set.

Property	Value
PidLidFlagRequest	SHOULD be a non-zero length string. "Follow up" SHOULD be used as the default if one is not specified explicitly by the user. Only set if PidLidFlagRequest does not already exist.
PidLidFlagString	See section 2.2.1.10 <29>Only set if PidLidFlagString does not already exist.
PidLidValidFlagStringProof	Equal to PidTagMessageDeliveryTime of the Message object <30>. Only set if PidLidValidFlagStringProof does not already exist.

For all Message objects that are not meeting-related objects or task objects, the following properties SHOULD also be set.

Property	Value
PidTagFlagStatus	followupFlagged (0x00000002).
PidTagFlagCompleteTime	Not present.
PidTagReplyRequested	0x01 <31>.
PidTagResponseRequested	0x01 <32>.

3.1.4.1.4 Setting a Complete Flag

The following properties SHOULD be set to these values on any Message object when marking a flagged Message object complete <33>.

Property	Value
PidTagFollowupIcon	Not present.
PidTagToDoItemFlags	todoTimeFlagged (0x00000001) bit set to 1.
PidLidToDoOrdinalDate	Equal to the current time in UTC if PidLidToDoOrdinalDate does not already exist.
PidLidToDoSubOrdinal	"5555555" if PidLidToDoSubOrdinal does not already exist.
PidLidTaskDateCompleted	Equal to the current date.
PidLidTaskComplete	0x01
PidLidTaskStatus	0x00000002
PidLidPercentComplete	1.0

For all Message objects that are not Task objects, the following properties SHOULD also be set.

Property	Value
PidLidToDoTitle	Equal to PidTagNormalizedSubject if PidLidToDoTitle does not already exist.

For all Message objects that are not meeting-related objects <34>, the following properties SHOULD be set with the following values:

Property	Value
PidLidFlagRequest	SHOULD be a non-zero length string. "Follow up" SHOULD be used as the default if one is not explicitly specified by the user. Set only if PidLidFlagRequest does not already exist.
PidLidFlagString	See section 2.2.1.10 <35>. Only set if PidLidFlagString does not already exist.
PidLidValidFlagStringProof	Equal to PidTagMessageDeliveryTime of the Message object <36> .Only set if PidLidValidFlagStringProof does not already exist.

For all Message objects that are not meeting-related objects or Task objects, the following properties SHOULD also be set.

Property	Value
PidTagFlagStatus	followupComplete (0x00000001).
PidTagFlagCompleteTime	Current time in UTC.
PidTagReplyRequested	0x00 <37>.
PidTagResponseRequested	0x00 <38>.

3.1.4.1.5 Setting a Recipient Flag

Recipient flags are set on Draft Message objects and are basic flags when they arrive in the recipients' mailbox. A message flagged for its recipients has the following properties set.

Property	Value
PidTagFlagStatus	Set to followupFlagged (0x00000002).
PidTagReplyRequested	SHOULD be set to 0x01 <39>.
PidTagResponseRequested	SHOULD be set to 0x01 <40>.
PidLidFlagRequest	SHOULD be a non-zero length string. "Follow up" SHOULD be used as the default if none explicitly specified by the user.
PidLidFlagString	See section 2.2.1.10 <41>.
PidLidTaskStatus	SHOULD be set to 0x00000000 <42>.
PidLidPercentComplete	SHOULD be set to 0.0 <43>.
PidLidTaskComplete	SHOULD be set to 0x00 <44>.

If the client supports sender flags, then PidTagSwappedToDoStore is set as specified in section 2.2.1.8 to trigger the post-transmit processing specified in section 3.1.4.3.

3.1.4.1.6 Setting a Sender Flag

On a Draft Message object, a client can set both sender flag properties and recipient flag properties <45>. The recipient flag properties are set in the **primary flag storage location**, as specified in section 3.1.4.1.5; the sender flag properties are set in the secondary flag storage location in PidTagSwappedToDoData, and PidTagSwappedToDoStore is set, as specified in section 2.2.1.8, to trigger the post-transmit processing, as specified in section 3.1.4.3.

3.1.4.2 Clearing a Flag on a Message Object

3.1.4.2.1 Clearing a Flag on a Meeting-Related Object

Clearing a flag on a meeting-related object is identical to clearing a flag on a message except that the following properties MUST NOT be deleted or altered: PidTagFlagStatus, PidTagFlagCompleteTime, PidTagReplyRequested, PidTagResponseRequested, PidLidFlagRequest, PidLidFlagString, and PidLidValidFlagStringProof.

3.1.4.2.2 Clearing a Flag on a Task Object

Flagging a task is a shortcut for the user to change the values of PidLidTaskStartDate and PidLidTaskDueDate. A client can, in this way, always view an uncompleted task as having a time flag. Because of this, clearing a flag on a task can be viewed as equivalent to deleting the Task object.

3.1.4.2.3 Clearing a Flag on Other Message Objects

To clear a flag from a Message object, the properties specified in section 2.2 SHOULD be deleted with the following **exceptions**. These properties are set to the following values.

Property	Value
PidTagToDoItemFlags	Set the todoTimeFlagged (0x00000001) bit to 0.

The following properties can be set to the following values instead of being deleted from the object.

Property	Value
PidLidTaskComplete	0x00 <46>
PidLidTaskStatus	0x00000000 <47>
PidLidPercentComplete	0.0 <48>
PidTagReplyRequested	0x00 <49>.
PidTagResponseRequested	0x00<50>.
PidLidFlagRequest	Zero-length string. <51>
PidLidFlagString	0x00000000 <52>
PidLidToDoOrdinalDate	4501/01/01 00:00:00.000 <53>
PidLidToDoSubOrdinal	Zero-length string. <54>

3.1.4.3 Post-Transmit Processing of a Flagged Message

Once a message that has a value set for the PidTagSwappedToDoStore (section 2.2.1.8) property has been sent, a client that supports sender flags or sender reminders takes the following actions. <55>

If the value of PidTagSwappedToDoStore matches the value of PidTagStoreEntryId of the Message object, it swaps the contents of PidTagSwappedToDoData and the primary flag storage location as shown by the mapping in section 2.2.1.7 with the exception that, if PidTagToDoItemFlags contains the value todoRecipientFlagged, signifying that the message was flagged for recipients as described in section 2.2.1.6, then the dwToDoItemFlags member of PidTagSwappedToDoData is set to todoTimeFlagged. After the operation, the previous contents of the primary flag storage location are now in the secondary flag storage location and vice versa. In addition, the value of PidLidValidFlagStringProof is set equal to the value of PidTagMessageDeliveryTime.

If the value of PidTagSwappedToDoStore does not match the value of PidTagStoreEntryId, the client clears the PidTagSwappedToDoData property.

In both cases, PidTagSwappedToDoStore is deleted.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7   Other Local Events

None.

4 Protocol Examples

All examples in this section assume that the user who is flagging the object resides in the Pacific Standard Time Zone (UTC -8). The following are descriptions of what a client does to accomplish the scenarios and the responses that a server returns.

Before flagging any Message objects, the client needs to ask the server to perform a mapping from **named properties** to **property identifiers**, using RopGetPropertyIDsFromNames.

Property	Property set GUID	Name or ID
PidLidFlagRequest	{00062008-0000-0000-C000-000000000046}	0x8530
PidLidFlagString	{00062008-0000-0000-C000-000000000046}	0x85C0
PidLidValidFlagStringProof	{00062008-0000-0000-C000-000000000046}	0x85BF
PidLidToDoTitle	{00062008-0000-0000-C000-000000000046}	0x85A4
PidLidToDoOrdinalDate	{00062008-0000-0000-C000-000000000046}	0x85A0
PidLidToDoSubOrdinal	{00062008-0000-0000-C000-000000000046}	0x85A1
PidLidTaskStartDate	{00062003-0000-0000-C000-000000000046}	0x8104
PidLidTaskDueDate	{00062003-0000-0000-C000-000000000046}	0x8105
PidLidCommonStart	{00062008-0000-0000-C000-000000000046}	0x8516
PidLidCommonEnd	{00062008-0000-0000-C000-000000000046}	0x8517
PidLidTaskDateCompleted	{00062003-0000-0000-C000-000000000046}	0x810F
PidLidTaskComplete	{00062003-0000-0000-C000-000000000046}	0x811C
PidLidTaskStatus	{00062003-0000-0000-C000-000000000046}	0x8101
PidLidPercentComplete	{00062003-0000-0000-C000-000000000046}	0x8102
PidLidReminderSet	{00062008-0000-0000-C000-000000000046}	0x8503
PidLidReminderDelta	{00062008-0000-0000-C000-000000000046}	0x8501
PidLidReminderTime	{00062008-0000-0000-C000-000000000046}	0x8502
PidLidReminderSignalTime	{00062008-0000-0000-C000-000000000046}	0x8560

The server responds with the following identifiers, which will be used in the examples that follow (the actual identifiers are at the discretion of the server).

Property	Property Identifier
PidLidFlagRequest	0x802A
PidLidFlagString	0x83C0
PidLidValidFlagStringProof	0x83CA

Property	Property Identifier
PidLidToDoTitle	0x8018
PidLidToDoOrdinalDate	0x830D
PidLidToDoSubOrdinal	0x830E
PidLidTaskStartDate	0x8143
PidLidTaskDueDate	0x8144
PidLidCommonStart	0x81BC
PidLidCommonEnd	0x81BB
PidLidTaskDateCompleted	0x8149
PidLidTaskComplete	0x8148
PidLidTaskStatus	0x8145
PidLidPercentComplete	0x8146
PidLidReminderSet	0x8004
PidLidReminderDelta	0x81FF
PidLidReminderTime	0x8005
PidLidReminderSignalTime	0x8006

4.1 Color-Flagged Object

Ryan Gregg has a message in his Inbox related to the "Woodgrove Bank" account, which he has associated in his mind to the orange flag. He uses the default request string designated by the client, which is "Follow up" <56>.

The client sends a RopOpenMessage request and waits for the server to respond. The server response contains a handle to the Message object.

The client sends a RopGetPropertiesSpecific request to retrieve some properties of the Message object.

Property	Property ID	Property type
PidTagMessageDeliveryTime	0x0E06	0x0040 (**PtypTime**)

The client receives a **ROP response buffer** from the server with the requested values.

Property	Property ID	Property type	Value
PidTagMessageDeliveryTime	0x0E06	0x0040 (**PtypTime**)	2008/02/11 22:41:24.765

The client then sends a RopSetProperties request to set the flagging properties.

Property	Property ID	Property type	Value
PidTagFlagStatus	0x1090	0x0003 (**PtypInteger32**)	0x00000002
PidTagFollowupIcon	0x1095	0x0003 (**PtypInteger32**)	0x00000002
PidTagReplyRequested	0x0C17	0x000B (**PtypBoolean**)	0x01
PidTagResponseRequested	0x0063	0x000B (**PtypBoolean**)	0x01
PidLidFlagRequest	0x802A	0x001F (**PtypString**)	Follow up
PidLidFlagString	0x83C0	0x0003 (**PtypInteger32**)	0x00000003
PidLidValidFlagStringProof	0x83CA	0x0040 (**PtypTime**)	2008/02/11 22:41:24.765

Finally, the client then sends a RopSaveChangesMessage request to persist the object on the server, and a RopRelease request to release the object.

The above properties are all that is strictly necessary to color flag an object. Clients can set the other properties discussed in section 2.2.1.7 to appropriate values <57>, if desired.

4.2   Time Flagged Object

Kendall Keil has a message in his Inbox with the subject "Contoso Project" and he wants to be sure to remember to follow up on that message starting on 2008/02/11, and needs to be done by 2008/02/12. He uses the default request string designated by the client, which is "Follow up". He flags the item on 2008/02/11 22:16:28.177 (UTC).

As before, the client first retrieves a handle to the Message object using RopOpenMessage.

Besides a handle, the following relevant data is returned from the response to the RopOpenMessage request.

Property	Property ID	Property type	Value
PidTagNormalizedSubject	0x0E1D	0x001F (**PtypString**)	Contoso Project

The client sends a RopGetPropertiesSpecific request to retrieve the state of the Message object.

Property	Property ID	Property type
PidTagMessageDeliveryTime	0x0E06	0x0040 (**PtypTime**)

The client receives a ROP response buffer from the server with the requested values.

Property	Property ID	Property type	Value
PidTagMessageDeliveryTime	0x0E06	0x0040 (**PtypTime**)	2008/02/11 22:41:24.765

The client then sends a RopSetProperties request to set the flagging properties.

Property	Property ID	Property type	Value
PidTagFlagStatus	0x1090	0x0003 (**PtypInteger32**)	0x00000002

Property	Property ID	Property type	Value
PidTagFollowupIcon	0x1095	0x0003 (**PtypInteger32**)	0x00000006
PidTagReplyRequested	0x0C17	0x000B (**PtypBoolean**)	0x01
PidTagResponseRequested	0x0063	0x000B (**PtypBoolean**)	0x01
PidTagToDoItemFlags	0x0E2B	0x0003 (**PtypInteger32**)	0x00000001
PidLidFlagRequest	0x802A	0x001F (**PtypString**)	Follow up
PidLidFlagString	0x83C0	0x0003 (**PtypInteger32**)	0x00000003
PidLidValidFlagStringProof	0x83CA	0x0040 (**PtypTime**)	2008/02/11 22:41:24.765
PidLidToDoTitle	0x8018	0x001F (**PtypString**)	Contoso Project
PidLidToDoOrdinalDate	0x830D	0x0040 (**PtypTime**)	2008/02/11 22:16:28.177
PidLidToDoSubOrdinal	0x830E	0x001F (**PtypString**)	5555555
PidLidTaskStartDate	0x8143	0x0040 (**PtypTime**)	2008/02/11 00:00:00.000
PidLidTaskDueDate	0x8144	0x0040 (**PtypTime**)	2008/02/12 00:00:00.000
PidLidCommonStart	0x81BC	0x0040 (**PtypTime**)	2008/02/11 08:00:00.000
PidLidCommonEnd	0x81BB	0x0040 (**PtypTime**)	2008/02/12 08:00:00.000

Finally, the client then sends a RopSaveChangesMessage request to persist the object on the server, and a RopRelease request to release the object.

4.3 Completed Object

Kendall Keil is now finished with the work item associated with the message in the previous example. As before, the client first retrieves a handle to the Message object using RopOpenMessage.

The client sends a RopSetProperties request to set the flagging properties.

Property	Property ID	Property ID	Value
PidTagFlagStatus	0x1090	0x0003 (**PtypInteger32**)	0x00000001
PidTagFlagCompleteTime	0x1091	0x0040 (**PtypTime**)	2008/02/11 22:23:00.000
PidTagReplyRequested	0x0C17	0x000B (**PtypBoolean**)	0x00
PidTagResponseRequested	0x0063	0x000B (**PtypBoolean**)	0x00
PidLidTaskDateCompleted	0x8149	0x0040 (**PtypTime**)	2008/02/11 08:00:00.000
PidLidTaskComplete	0x8148	0x000B (**PtypBoolean**)	0x01
PidLidTaskStatus	0x8145	0x0003 (**PtypInteger32**)	0x00000002
PidLidPercentComplete	0x8146	0x0005 (**PtypFloating64**)	1.0
PidLidReminderSet	0x8004	0x000B (**PtypBoolean**)	0x00

The application sends a RopDeletePropertiesNoReplicate request.

Property	Property ID	Property ID
PidTagFollowupIcon	0x1095	0x0003 (**PtypInteger32**)

Finally, the client sends a RopSaveChangesMessage request to persist the object on the server, and a RopRelease request to release the object.

4.4 Flagging a Draft Message Object for the Sender and Recipient

Randy Byrne is planning to send a contract to a customer on March 7, 2008. He wants his co-worker, Marina Dukhon, to review the contract before he sends it out. He sends a message to Marina with a sender flag that has a due date of 2008/03/07 with a request string of "Forward", and a recipient flag to Marina with a reminder on March 6, 2008 at 4:00 P.M. local time and a request string of "Review".

As before, the client first retrieves a handle to the Message object using RopOpenMessage.

The client sends a RopSetProperties request to set the flagging properties.

Property	Property ID	Property type	Value
PidTagFlagStatus	0x1090	0x0003 (**PtypInteger32**)	0x00000002
PidTagReplyRequested	0x0C17	0x000B (**PtypBoolean**)	0x01
PidTagResponseRequested	0x0063	0x000B (**PtypBoolean**)	0x01
PidTagReplyTime	0x0030	0x0040 (**PtypTime**)	2008/03/07 00:00:00.000
PidTagToDoItemFlags	0x0E2B	0x0003 (**PtypInteger32**)	0x00000008
PidTagSwappedToDoData	0x0E2D	0x0102 (**PtypBinary**)	See below
PidTagSwappedToDoStore	0xE2C	0x0102 (**PtypBinary**)	See below
PidLidFlagRequest	0x802A	0x001F (**PtypString**)	Review
PidLidFlagString	0x83C0	0x0003 (**PtypInteger32**)	0x0000000A
PidLidToDoTitle	0x8018	0x001F (**PtypString**)	Review
PidLidTaskStatus	0x8147	0x0003 (**PtypInteger32**)	0x00000000
PidLidPercentComplete	0x8146	0x0005 (**PtypFloating64**)	0.0
PidLidTaskComplete	0x8148	0x000B (**PtypBoolean**)	0x00
PidLidReminderSet	0x8004	0x000B (**PtypBoolean**)	0x01
PidLidReminderDelta	0x81FF	0x0003 (**PtypInteger32**)	0x00000000
PidLidReminderTime	0x8005	0x0040 (**PtypTime**)	2008/03/07 00:00:00.000
PidLidReminderSignalTime	0x8006	0x0040 (**PtypTime**)	2008/03/07 00:00:00.000

The value of PidTagSwappedToDoStore is set to the value of PidTagStoreEntryId of the Message object.

The value of PidTagSwappedToDoData is:

```
000: 01 00 00 00 79 00 00 00 01 00 00 00 46 00 6F 00
010: 72 00 77 00 61 00 72 00 64 00 00 00 00 00 00 00
020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
200: 00 00 00 00 00 00 00 00 00 00 00 00 E0 80 E9 5A
210: 60 C0 C3 0C 00 00 00 00 00 00 00 00
```

This corresponds to the following values.

Member	Value
ulVersion	0x00000001
dwFlags	0x00000079
dwToDoItem	0x00000001
wszFlagTo	Forward
rtmStartDate	0x5AE980E0 (None)
rtmDueDate	0x0CC3C060 (2008/03/07 00:00:00.000)
rtmReminder	0x00000000

Release: Friday, February 5, 2010

Member	Value
fReminderSet	0x00000000

Finally, the client then sends a RopSubmitMessage request to send the message to the intended recipients, and a RopRelease request to release the object.

When the above message is sent and is found by the client in Randy's mailbox at March 3, 2008 21:03:29.438, the value of PidTagSwappedToDoStore on the Message object is compared to the value of PidTagStoreEntryId of the Message object and, if they are equal, the contents of the primary and secondary flag storage locations are swapped. The client retrieves a handle to the object by sending a RopOpenMessage request, and gets the relevant flagging properties by sending a RopGetPropertiesSpecific request. The values of the properties returned would be identical to the values set above, except for the following, which is changed by the server during Message delivery.

Property	Property ID	Property type	Value
PidTagMessageDeliveryTime	0x0E06	0x0040 (**PtypTime**)	2008/03/03 21:03:00.000

The client sends a RopSetProperties request to perform the swap.

Property	Property ID	Property type	Value
PidTagFlagStatus	0x1090	0x0003 (**PtypInteger32**)	0x00000002
PidTagFollowupIcon	0x1095	0x0003 (**PtypInteger32**)	0x00000006
PidTagReplyRequested	0x0C17	0x000B (**PtypBoolean**)	0x01
PidTagResponseRequested	0x0063	0x000B (**PtypBoolean**)	0x01
PidTagToDoItemFlags	0x0E2B	0x0003 (**PtypInteger32**)	0x00000001
PidTagSwappedToDoData	0x0E2D	0x0102 (**PtypBinary**)	See below.
PidLidFlagRequest	0x802A	0x001F (**PtypString**)	Forward
PidLidFlagString	0x83C0	0x0003 (**PtypInteger32**)	0x00000000
PidLidValidFlagStringProof	0x83CA	0x0040 (**PtypTime**)	2008/03/03 21:03:00.000
PidLidToDoTitle	0x8018	0x001F (**PtypString**)	Contoso Project
PidLidToDoOrdinalDate	0x830D	0x0040 (**PtypTime**)	2008/03/03 21:03:29.438
PidLidToDoSubOrdinal	0x830E	0x001F (**PtypString**)	5555555
PidLidTaskDueDate	0x8144	0x0040 (**PtypTime**)	2008/03/07 00:00:00.000
PidLidCommonEnd	0x81BB	0x0040 (**PtypTime**)	2008/03/07 08:00:00.000
PidLidTaskComplete	0x8148	0x000B (**PtypBoolean**)	0x00
PidLidTaskStatus	0x8145	0x0003 (**PtypInteger32**)	0x00000000
PidLidPercentComplete	0x8146	0x0005 (**PtypFloating64**)	0.0
PidLidReminderSet	0x8004	0x000B (**PtypBoolean**)	0x00

The value of PidTagSwappedToDoData is:

```
000: 01 00 00 00 F9 00 00 00 01 00 00 00 52 00 65 00
010: 76 00 69 00 65 00 77 00 00 00 00 00 00 00 00 00
020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
200: 00 00 00 00 00 00 00 00 00 00 00 00 E0 80 E9 5A
210: E0 80 E9 5A 60 C0 C3 0C 01 00 00 00
```

This corresponds to the following values.

Member	Value
ulVersion	0x00000001
dwFlags	0x000000F9
dwToDoItem	0x00000001
wszFlagTo	Review
rtmStartDate	0x5AE980E0 (None)
rtmDueDate	0x5AE980E0 (None)
rtmReminder	0x0CC3C060 (2008/03/07 00:00:00.000)
fReminderSet	0x00000001

The application sends a RopDeletePropertiesNoReplicate request.

Property	Property ID	Property type
PidTagSwappedToDoStore	0xE2C	0x0102 (**PtypBinary**)
PidLidTaskStartDate	0x8143	0x0040 (**PtypTime**)
PidLidCommonStart	0x81BC	0x0040 (**PtypTime**)
PidLidTaskDateCompleted	0x8149	0x0040 (**PtypTime**)
PidLidReminderTime	0x8005	0x0040 (**PtypTime**)
PidLidReminderSignalTime	0x8006	0x0040 (**PtypTime**)
PidTagReplyTime	0x0030	0x0040 (**PtypTime**)

Finally, the client sends a RopSaveChangesMessage request to persist the object on the server, and a RopRelease request to release the object.

5   Security

5.1   Security Considerations for Implementers

There are no security considerations specific to the Informational Flagging protocol. General security considerations pertaining to the underlying transport apply, as specified in [MS-OXCMSG] and [MS-OXOMSG].

5.2   Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following product versions. References to product versions include released service packs.

- Microsoft Office Outlook 2003

- Microsoft Exchange Server 2003

- Microsoft Office Outlook 2007

- Microsoft Exchange Server 2007

- Microsoft Outlook 2010

- Microsoft Exchange Server 2010

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. The new behavior also applies to subsequent service packs of the product unless otherwise specified.

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

<1> Section 2.2.1.1: Outlook 2003 does provide a way for the user to set the value of PidTagFlagStatus on a Task object.

<2> Section 2.2.1.2: Outlook 2003 does provide a way for the user to set the value of PidTagFollowupIcon on a Task object.

<3> Section 2.2.1.3: Outlook 2003 does provide a way for the user to set the value of PidTagFlagCompleteTime on a Task object.

<4> Section 2.2.1.4: Outlook 2003 does provide a way for the user to set the value of PidTagReplyRequested and PidTagResponseRequested on a Task object.

<5> Section 2.2.1.4: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagReplyRequested or PidTagResponseRequested.

<6> Section 2.2.1.4: In Outlook 2007, when swapping the contents of the primary and secondary flagging storage location, the properties PidTagReplyRequested and PidTagResponseRequested do not get updated according to the new flagging state of the primary flagging storage location to the correct value.

<7> Section 2.2.1.6: Outlook 2003 and Exchange 2003 do not read or write this flag.

<8> Section 2.2.1.6: In Exchange 2007 and Exchange 2010, if an unflagged object is flagged complete (without first being time flagged), the bit todoTimeFlagged (0x00000001) of PidTagToDoItemFlags is not set.

<9> Section 2.2.1.7: Sender flags are not supported in Outlook 2003 or Exchange 2003.

<10> Section 2.2.1.9: Exchange 2003, Exchange 2007, and Exchange 2010 do not support user-specifiable text. Instead, the servers always use "Follow Up".

<11> Section 2.2.1.9: Exchange 2003, Exchange 2007, and Exchange 2010 do not understand PidLidFlagString or PidLidValidFlagStringProof, so the value of PidLidFlagRequest is always used.

<12> Section 2.2.1.10: Exchange 2003, Exchange 2007, and Exchange 2010 do not read or write PidLidFlagString. Also, there are user interface elements within Outlook that allow an item to be flagged without this property being set.

<13> Section 2.2.1.10: Flagging meeting-related objects with PidLidFlagString will result in undefined behavior.

<14> Section 2.2.1.11: Flagging meeting related objects with PidLidValidFlagStringProof will result in undefined behavior.

<15> Section 2.2.1.11: Exchange 2003, Exchange 2007, and Exchange 2010 do not read or write PidLidValidFlagStringProof. Also, there are user interface elements within Outlook that allow an item to be flagged without this property being set.

<16> Section 2.2.1.11: Exchange 2003, Exchange 2007, and Exchange 2010 do not read or write PidLidValidFlagStringProof.

<17> Section 2.2.1.12: Outlook 2007 substitutes the concatenated values of PidTagSubjectPrefix and PidTagNormalizedSubject when displaying a Message object without a value in the PidLidToDoTitle property.

<18> Section 2.2.1.13: Outlook 2003 and Exchange 2003 do not read or write PidLidToDoOrdinalDate.

<19> Section 2.2.3.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not set or clear PidTagReplyTime.

<20> Section 3.1.4.1: The default Outlook user interface does not permit users to flag Appointment objects, Journal objects, or sticky notes, but it is possible to manipulate the UI in non-standard ways in order to set flag-related properties on such objects. Flagging Appointment objects, Journal objects, or sticky notes will result in undefined behavior.

<21> Section 3.1.4.1.1: Users can apply color flags only in Outlook 2003 or Exchange 2003.

<22> Section 3.1.4.1.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidFlagString.

<23> Section 3.1.4.1.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidValidFlagStringProof.

<24> Section 3.1.4.1.1: Outlook 2003 writes these properties on Task objects as well.

<25> Section 3.1.4.1.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagReplyRequested.

<26> Section 3.1.4.1.1: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagResponseRequested.

<27> Section 3.1.4.1.2: Except for receiving a recipient flag, (see paragraph 3.1.4.1.5), basic flags can be set only in Outlook 2003.

<28> Section 3.1.4.1.3: Users cannot apply time flags in Outlook 2003 or Exchange 2003.

<29> Section 3.1.4.1.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidFlagString.

<30> Section 3.1.4.1.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidValidFlagStringProof.

<31> Section 3.1.4.1.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagReplyRequested.

<32> Section 3.1.4.1.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagResponseRequested.

<33> Section 3.1.4.1.4: In addition to the exceptions as noted for individual properties, Outlook 2003 and Exchange 2003 do not set the following properties when marking a Message object complete: PidTagToDoItemFlags, PidLidToDoTitle, PidLidToDoOrdinalDate, PidLidToDoSubOrdinal, PidLidTaskDateCompleted, PidLidTaskComplete, PidLidTaskStatus, and PidLidPercentComplete. Note that since Outlook 2003 and Exchange 2003 also do not set PidTagFlagStatus for meeting-related objects, those objects cannot be flagged complete in Outlook 2003 and Exchange 2003.

<34> Section 3.1.4.1.4: Outlook 2003 does not support time flags. PidTagFlagStatus cannot be set on a meeting-related object; therefore, Outlook 2003 does not support marking a meeting-related object as complete.

<35> Section 3.1.4.1.4: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidFlagString.

<36> Section 3.1.4.1.4: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidValidFlagStringProof.

<37> Section 3.1.4.1.4: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagReplyRequested.

<38> Section 3.1.4.1.4: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagResponseRequested.

<39> Section 3.1.4.1.5: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagReplyRequested.

<40> Section 3.1.4.1.5: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidTagResponseRequested.

<41> Section 3.1.4.1.5: Exchange 2003, Exchange 2007, and Exchange 2010 do not set PidLidFlagString.

<42> Section 3.1.4.1.5: Outlook 2003 and Exchange 2003, Exchange 2007, and Exchange 2010 do not set this property.

<43> Section 3.1.4.1.5: Outlook 2003 and Exchange 2003, Exchange 2007, and Exchange 2010 do not set this property.

<44> Section 3.1.4.1.5: Outlook 2003 and Exchange 2003, Exchange 2007, and Exchange 2010 do not set this property.

<45> Section 3.1.4.1.6: Sender flags are not supported in Outlook 2003. Only recipient flags can be set in Outlook 2003.

<46> Section 3.1.4.2.3: Exchange 2007 and Exchange 2010 delete the. PidLidTaskComplete property.

<47> Section 3.1.4.2.3: Exchange 2007 and Exchange 2010 delete the PidLidTaskStatus property.

<48> Section 3.1.4.2.3: Exchange 2007 and Exchange 2010 delete the PidLidPercentComplete property.

<49> Section 3.1.4.2.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set nor clear the PidTagReplyRequested property.

<50> Section 3.1.4.2.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set or clear the PidTagResponseRequested property.

<51> Section 3.1.4.2.3: Exchange 2007 and Exchange 2010 delete the PidLidFlagRequest property.

<52> Section 3.1.4.2.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set or clear the PidLidFlagString property.

<53> Section 3.1.4.2.3: Exchange 2003, Exchange 2007, and Exchange 2010 do not set or clear the PidLidToDoOrdinalDate property.

<54> Section 3.1.4.2.3: Exchange 2003, Exchange 2007, and Exchange 2010 do neither; they do not set or clear the PidLidToDoSubOrdinal property.

<55> Section 3.1.4.3: Outlook 2003 and Exchange 2003, Exchange 2007, and Exchange 2010 do not support sender flags, and thus do not support this portion of the protocol.

<56> Section 4.1: This example applies only to Outlook 2003 and Exchange 2003.

<57> Section 4.1: Because time flags were introduced in Outlook 2007, and color flags are supported in Outlook 2003 and discontinued in Outlook 2007, this color flagging example shows what properties are set by Outlook 2003. By following this example, some properties could end up unsynchronized with each other (i.e. applying a color flag on an object that was previously flagged complete by Outlook 2007 following this example would not reset the value of PidLidTaskComplete). Other clients that only support color flags can choose to go further than this example, and set or reset all of the other properties to the appropriate values.

7 Change Tracking

This section identifies changes made to [MS-OXOFLAG] protocol documentation between November 2009 and February 2010 releases. Changes are classed as major, minor, or editorial.

Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.

- An extensive rewrite, addition, or deletion of major portions of content.

- A protocol is deprecated.

- The removal of a document from the documentation set.

- Changes made for template compliance.

Minor changes do not affect protocol interoperability or implementation. Examples are updates to fix technical accuracy or ambiguity at the sentence, paragraph, or table level.

Editorial changes apply to grammatical, formatting, and style issues.

No changes means that the document is identical to its last release.

Major and minor changes can be described further using the following revision types:

- New content added.

- Content update.

- Content removed.

- New product behavior note added.

- Product behavior note updated.

- Product behavior note removed.

- New protocol syntax added.

- Protocol syntax updated.

- Protocol syntax removed.

- New content added due to protocol revision.

- Content updated due to protocol revision.

- Content removed due to protocol revision.

- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.

- Protocol syntax removed due to protocol revision.

- New content added for template compliance.

- Content updated for template compliance.

- Content removed for template compliance.

- Obsolete document removed.

Editorial changes always have the revision type "Editorially updated."

Some important terms used in revision type descriptions are defined as follows:

Protocol syntax refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.

Protocol revision refers to changes made to a protocol that affect the bits that are sent over the wire.

Changes are listed in the following table. If you need further information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
2.2.1.7 PidTagSwappedToDoData	52047 Added a packet diagram to fully represent the information.	Y	Content update.
3.1.4.1.5 Setting a Recipient Flag	52050 Modified Product Behavior notes to remove ambiguity.	N	Content update.
4 Protocol Examples	52049 Clarified ambiguous wording.	N	Editorially updated.

8 Index