[MS-OXORMDR]: Reminder Settings 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.
- **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			
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.
Microsoft Corporation	June 27, 2008	1.0	Initial Release.
Microsoft Corporation	August 6, 2008	1.01	Revised and edited technical content.
Microsoft Corporation	September 3, 2008	1.02	Updated references.
Microsoft Corporation	December 3, 2008	1.03	Updated IP notice.
Microsoft Corporation	April 10, 2009	2.0	Updated technical content for new product releases.

Table of Contents

1 Introduction

This document specifies a protocol for discovering and acting upon **Message objects** that have a deadline. This includes upcoming (or past) appointments, tasks, messages, or contacts for which follow-up is necessary, or any other Message object for which the properties specified in this document have been set.

The Reminder Settings protocol specifies the following:

- Methods by which a client can add, remove, or modify reminders on a Message object.
- Methods by which a client can dismiss or snooze a reminder.
- Details that allow the client to process overdue reminders.

1.1 Glossary

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

appointment
binary large object (BLOB)
Calendar object
Draft Message object
exception
Exception Attachment object
Exception Embedded Message object
folder
handle
meeting-related object
message
Message object
property
recurrence pattern
Recurring Calendar object
recurring series
recurring task
reminder
search folder
single instance
signal time
special folder
store
table
Task object

UTC

The following data types are defined in [MS-OXCDATA]:

PtypBinary
PtypBoolean
PtypInteger32
PtypString
PtypTime

The following terms are specific to this document:

active reminder: A **reminder** that is enabled on an object. An **active reminder** is either pending or overdue, depending on whether the **signal time** has passed.

dismiss: To disable an **overdue reminder**. After a **reminder** is **dismissed**, it is no longer considered overdue and is, therefore, no longer signaled/displayed to the user and/or any agents acting on behalf of the user.

due time: The time after which a user is considered late, such as the start time of an appointment or the time at which a work item is expected to be completed.

full domain: See **full reminder domain**.

full reminder domain: The maximum scope a client can use when searching for objects that have **reminders** enabled. A **full reminder domain** client is a client that includes all folders when searching for objects with an enabled **reminder** (for example, a recursive search starting at the top of personal folders, as specified in [MS-OXOSFLD]). The **full reminder domain** typically includes all such folders except the following: Deleted Items, Junk E-mail, Drafts, Outbox, Conflicts, Local Failures, Server Failures, and Sync Issues. (See [MS-OXOSFLD] for details about how to identify these folders.)

minimal domain: See **minimal reminder domain**.

minimal reminder domain: The smallest scope a client is allowed to use when searching for objects that have an **active reminder**. The **minimal reminder domain** includes the following folders: Inbox, primary Contacts, primary Calendar, and primary Tasks. (See [MS-OXOSFLD] for details about how to identify these folders.) The **minimal reminder domain** does not include sub-folders.

overdue reminder: An **active reminder** whose **signal time** has passed.

pending reminder: An **active reminder** whose **signal time** is in the future.

recurrence BLOB: The **BLOB** encoding of a **recurrence pattern**, a **recurrence range**, and **exceptions**.

recurrence range: The range of time for which a **recurrence pattern** continues.

Recurring object: A **Recurring Calendar object** or a **Recurring Task object**.

Recurring Task object: A **Task object** that represents a **recurring task**.

reminder domain: The set of folders that are searched for objects that have an **active reminder**.

reminder properties: A set of properties that specify the attributes of a **reminder**. These attributes include the time at which and the method by which the **reminder** is to be signaled or displayed.

reminder queue: A sorted list of objects in the **reminder domain** that have been stamped with properties that imply they could have an **active reminder**.

single instance Calendar object: A **Calendar object** that represents a **single instance**.

single instance object: A **single instance Calendar object** or a **single instance Task object**, or an object of a type that does not support recurrences.

single instance Task object: A **Task object** that does not represent a **recurring task**.

snooze: To delay an **overdue reminder** by a specified interval of time at the end of which it will become an **overdue reminder** once again.

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-OXCDATA] Microsoft Corporation, "Data Structures Protocol Specification", June 2008.

[MS-OXCFOLD] Microsoft Corporation, "Folder Object Protocol Specification", 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-OXCSTOR] Microsoft Corporation, "Store Object Protocol Specification", June 2008.

[MS-OXCTABL] Microsoft Corporation, "Table 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-OXOCFG] Microsoft Corporation, "Configuration Information Protocol Specification", June 2008.

[MS-OXOFLAG] Microsoft Corporation, "Informational Flagging Protocol Specification", June 2008.

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

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

[MS-OXOSRCH] Microsoft Corporation, "Search Folder List Configuration 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 Specification", 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

This protocol allows the user or an agent who is acting on behalf of the user to associate a **reminder** with a given **Message object** with the intention that a client will somehow signal the user or agent when the **signal time** is reached.

1.3.1 Creating, Modifying, and Removing Reminders

The client creates, modifies, or removes **reminders** by setting **reminder properties** on objects, as specified in section 3.1.4. The client can specify such things as when to signal a reminder, the due time for the object about which the user is to be reminded, whether to play a sound when the reminder is signaled, which sound to play, and whether the reminder is enabled. On a **Draft Message object**, reminders for the recipient or recipients can be specified independently from those for the sender.

1.3.2 Processing Overdue Reminders

The client processes overdue **reminders**, as specified in section 3.1.4.5. Before signaling an **overdue reminder**, the client is responsible for deciding which reminders to ignore and which reminders to automatically **dismiss**.

1.3.3 Dismissing and Snoozing Reminders

The client can **dismiss** or **snooze overdue reminders**, as specified in sections 3.1.4.6 and 3.1.4.7.

1.4 Relationship to Other Protocols

The Reminder Setting protocol extends the Message and Attachment Object protocol and relies on an understanding of how to work with **folders**, **messages**, and **tables**. For details, see [MS-OXCDATA], [MS-OXPROPS], [MS-OXOMSG], [MS-OXCFOLD], [MS-OXCTABL], [MS-OXOSFLD], [MS-OXOSRCH], [MS-OXOCFG], [MS-OXOCAL], [MS-OXOTASK], and [MS-OXOFLAG].

1.5 Prerequisites/Preconditions

For requesting and configuring **reminders**, the Reminder Settings protocol assumes that the client has previously logged on to the server and has acquired a **handle** to the object for which it intends to create or update **reminder properties** [MS-OXCFOLD] [MS-OXCMSG].

For discovering **Message objects** that have reminders, the Reminder Settings protocol assumes that the client has previously logged on to the server and has acquired a handle to the mailbox **store** in which it intends to search[MS-OXCSTOR].

1.6 Applicability Statement

The Reminder Settings protocol is used to specify and raise time-specific notifications to a user or an agent who is acting on behalf of a user.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

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

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

This protocol uses the transport specified in [MS-OXCMSG].

2.2 Message Syntax

Reminders can be created and modified, on any type of **Message object**, by clients and servers. Except where noted, this section defines constraints to which both clients and servers adhere when operating on the reminder properties of message objects.

Clients operate on the **reminder properties** of Message objects by using the Message and Attachment Object protocol, as specified in [MS-OXCMSG]. How a server operates on the reminder properties of Message objects is implementation-dependent. The results of any such operation are exposed to clients in a manner that is consistent with the Reminder Settings protocol.

2.2.1 Properties Used to Specify and/or Decode Reminders

The following properties are used to specify the **reminder** for a given object. These properties are set by the client to ensure that the user is notified in the manner and at the time that is preferred.

2.2.1.1 PidLidReminderSet

Type: **PtypBoolean**.

Specifies whether a **reminder** is set on the object.

If a **Recurring Calendar object** has **PidLidReminderSet** set to **TRUE**, the client can override this value for exceptions. See the definition of **PidLidAppointmentRecur** in [MS-OXOCAL] for details.

If **PidLidReminderSet** is **FALSE** on a Recurring Calendar object, reminders are disabled for the entire series, including **exceptions**.

For **recurring task objects**, the **PidLidReminderSet** property cannot be overridden by exceptions (see [MS-OXOCAL] and [MS-OXOTASK] for details).

2.2.1.2 PidLidReminderSignalTime

Type: **PtypTime**.

Specifies the point in time when a **reminder** transitions from pending to overdue. This property MUST be set if **PidLidReminderSet** is **TRUE**. Clients MUST set the value in **UTC**.

This property is not overridden by **exceptions**; there is a single **signal time** per **Recurring Calendar object**.

In some cases, the value of this time property is not interpreted strictly as a UTC time. For more information, see [MS-OXOCAL].

2.2.1.3 PidLidReminderDelta

Type: **PtypInteger32**.

This property MUST be set on **Calendar objects** and specifies the interval, in minutes, between the time at which the reminder first becomes overdue and the start time of the Calendar object. For all non-Calendar objects, this property SHOULD be set to 0x00000000 and is ignored.

When a reminder is dismissed for one instance of a **Recurring Calendar object**, the value of this property is used in the calculation of the signal time for the next instance. See [MS-OXOCAL] for details about Calendar object creation.

2.2.1.4 PidLidReminderTime

Type: **PtypTime**.

For non-Calendar objects, this property specifies the initial **signal time**. For Calendar **objects**, this property represents the time after which the user would be late; that is, the start time of the **appointment**. Clients MUST set the value in **UTC**.

2.2.1.5 PidLidReminderOverride

Type: **PtypBoolean**.

If set to **TRUE**, specifies that the client SHOULD<1> respect the values **PidLidReminderPlaySound** and **PidLidReminderFileParameter**. Otherwise, a client can use default values in place of the values of **PidLidReminderPlaySound** and **PidLidReminderFileParameter**.

2.2.1.6 PidLidReminderPlaySound

Type: **PtypBoolean**.

If set to **TRUE**, specifies that the client SHOULD<2> play a sound when the **reminder** becomes overdue.

2.2.1.7 PidLidReminderFileParameter

Type: **PtypString**.

Specifies the filename of the sound<3> that a client SHOULD<4> play when the **reminder** for that object becomes overdue. If this property is not present, the client can use a default value.

2.2.1.8 PidTagReplyTime

Type: **PtypTime**.

Upon receipt of a **Message object**, specifies the **due time** that the sender wants, presumably for an associated work item, and is otherwise ignored. Clients MUST set the value in **UTC**. See [MS-OXOFLAG] for additional details.

2.2.1.9 PidLidReminderType

Type: **PtypInteger32**.

SHOULD NOT<5> be set, and MUST be ignored.

2.2.2 Shared Properties

These properties are set at **reminder** configuration time and/or processed and/or presented to the user (or an agent who is acting on behalf of the user) as described in the following sections.

2.2.2.1 Properties Shared with the Message and Attachment Object Protocol

The following properties are shared by this protocol and the Message and Attachment Object protocol [MS-OXCMSG]:

- **PidTagAttachDataObject**
- **PidTagMessageClass**
- **PidTagMessageFlags**

The semantics and accepted values are identical to those specified in [MS-OXCMSG].

2.2.2.2 Properties Shared with the Informational Flagging Protocol

The following properties are shared by this protocol and the Informational Flagging protocol [MS-OXOFLAG]:

- **PidTagSwappedToDoData**
- **PidTagSwappedToDoStore**

The semantics and accepted values are identical to those specified in [MS-OXOFLAG].

2.2.2.3 Properties Shared with the Task-Related Objects Protocol

The following properties are shared by this protocol and the Task-Related Objects protocol [MS-OXOTASK]:

- **PidLidTaskDueDate**
- **PidLidTaskRecurrence**
- **PidLidTaskResetReminder**

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

2.2.2.3.1 PidLidTaskDueDate

Type: **PtypTime**.

SHOULD<6> be set when creating a **reminder** on an object that is neither a **Task object** nor a **Calendar object**.

Setting **PidLidTaskDueDate** allows for a more intuitive "due-in" time when the reminder signals.

2.2.2.4 Properties Shared with the Appointment and Meeting Object Protocol

The following properties are shared by this protocol and the Appointment and Meeting Object protocol [MS-OXOCAL]:

- **PidLidAppointmentRecur**
- **PidLidAutoStartCheck**<7>
- **PidLidConferencingCheck**<8>

The semantics and accepted values are identical to those specified in [MS-OXOCAL].

2.2.3 Properties Used to Specify User Preferences that are Roamed on the Server

The following property is used to specify global state with respect to **reminders**. This property uses the protocol defined in [MS-OXOCFG] as a transport.

2.2.3.1 piReminderUpgradeTime

A 32-bit integer value (specified in number of minutes since midnight, January 1, 1601), stored in the Calendar Options Dictionary (see [MS-OXOCFG] for details), that specifies the first time a **full reminder domain** client was used on the mailbox. The time is interpreted in the user's current time zone, not **UTC**.

3 Protocol Details

Except where noted, this section defines behavioral constraints to which both clients and servers MUST adhere when operating on the reminder properties of **Message objects**.

3.1 Client and Server Details

The server fulfills the server role in the following protocols.

- Office Exchange Protocols Master Property List Specification [MS-OXPROPS]
- E-mail Object Protocol Specification [MS-OXOMSG]
- Folder Object Protocol Specification [MS-OXCFOLD]
- Table Object Protocol Specification [MS-OXCTABL]
- Special Folders Protocol Specification [MS-OXOSFLD]
- Search Folder List Configuration Protocol Specification [MS-OXOSRCH]
- Configuration Information Protocol Specification [MS-OXOCFG]
- Appointment and Meeting Object Protocol Specification [MS-OXOCAL]
- Task-Related Objects Protocol Specification [MS-OXOTASK]
- Informational Flagging Protocol Specification [MS-OXOFLAG]

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.

Messages with **active reminders** can be discovered by using a persistent **search folder** as specified in [MS-OXCFOLD]. Those messages can then be loaded into a list, referred to as the **reminder queue**, that holds the relevant information in memory for **pending** and/or **overdue reminders** and is sorted by the property **PidLidReminderSignalTime**.

Objects can be in one of the following states:

- No active reminder
- Pending reminder
- Overdue reminder

The **reminder queue** only contains **Message objects** that have a pending or overdue **reminder**.

3.1.2 Timers

None.

3.1.3 Initialization

3.1.3.1 Populating the Reminder Queue

3.1.3.1.1 Scope

Two approaches are typically used to scope the search for **active reminders** in the **reminder queue**: **minimal reminder domain** or **full reminder domain**.

Clients SHOULD use a **full domain** scope strategy <9>. Using multiple clients that use different scope strategies to access the same mailbox is not recommended<10>.

3.1.3.1.1.1 Requirements for Clients Using a Full Reminder Domain

Clients that use a **full domain** scope strategy MUST ensure that **piReminderUpgradeTime** is set. If **piReminderUpgradeTime** does not exist, the client MUST set it. The value SHOULD correspond to the current local time, to reflect the time of upgrade to the expanded scope strategy of the newer client. The property **piReminderUpgradeTime** SHOULD<11> only be set once. If **piReminderUpgradeTime** is already set, it SHOULD NOT be changed as it represents the first time an "upgraded" client was used.

3.1.3.1.2 Loading/Processing

When populating or updating the **reminder queue** with **Message objects** found in the **reminder domain**, the client SHOULD:

- Ignore certain objects. For details, see section 3.1.4.5.1.

- Auto-**dismiss** certain objects. For details, see section 3.1.4.5.2.
- Include any remaining objects that have **active reminders** (**pending** or **overdue reminders**).

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Setting a Reminder

In addition to the behaviors specified in this section, the client sets **PidLidReminderSet** to **TRUE** on an object to enable the signaling of a **reminder**. The client sets **PidLidReminderSignalTime** to the specified time the reminder becomes overdue, and sets **PidLidReminderTime**.

3.1.4.1.1 Reminders on Task Objects

The client sets **PidLidReminderTime** to the specified time at which the **reminder** becomes overdue.

3.1.4.1.2 Reminders on Neither Calendar nor Task Objects

The client sets **PidLidReminderTime** and SHOULD<12> set **PidTagReplyTime** to the specified **signal time**.

In summary, the client sets the following properties (or ensures that they are already set <13> for objects that are neither **Calendar objects** nor **Task objects**):

Property	Value
PidLidReminderSet	**TRUE**.
PidLidReminderTime	The specified **signal time**.
PidLidReminderSignalTime	The specified signal time.
PidTagReplyTime	The specified signal time.

3.1.4.1.3 Reminders on Calendar Objects

The client sets **PidLidReminderTime**, and the value SHOULD<14> correspond to the **appointment** start time (see [MS-OXOCAL] for details).

For **Calendar objects**, the client MUST set **PidLidReminderDelta** to reflect the specified interval, in minutes, between the signal time of the **reminder** and the start time of the Calendar object.

In summary, the client sets the following properties (or otherwise ensures that the properties are already set):

Property	Value
PidLidReminderSet	**TRUE**.
PidLidReminderTime	MUST be set, and SHOULD <15> be the appointment start time.
PidLidReminderDelta	The interval, in minutes, between signal time and the appointment start time.
PidLidReminderSignalTime	The signal time in **UTC**.

3.1.4.1.4 Reminders on Recurring Calendar Objects

For Recurring **Calendar objects**, the client also sets or modifies the **recurrence BLOB** (**PidLidAppointmentRecur**) on the **Recurring Calendar object**, and the **Exception Embedded Message Object** (**PidTagAttachDataObject**) on the **Exception Attachment object**, as specified in the following paragraphs.

To modify the **reminder** for specific instances in a recurring series, the client MUST create an **exception** if one does not already exist for that instance, and then modify the recurrence BLOB and Exception Embedded Message object, to store whichever properties are different from the Recurring Calendar object, such as **StartTime**, **ReminderDelta**, or **ReminderSet**. See [MS-OXOCAL] for details about the recurrence BLOB (**PidLidAppointmentRecur**), and the Exception Attachment object, and see [MS-OXCMSG] for details about accessing the Exception Embedded Message object, which is stored in **PidTagAttachDataObject**.

Setting **PidLidReminderSet** to **FALSE** on the Recurring Calendar object MUST take precedence over any reminders that are specified for exceptions. To disable reminders for the entire series, including all exceptions, set **PidLidReminderSet** to **FALSE** on the Recurring Calendar object.

If one or more instances, but not all instances, need a reminder, set **PidLidReminderSet** to **TRUE** on the Recurring Calendar object. Then set the ReminderSet field to **FALSE** in the recurrence BLOB for each instance that does not have a reminder, and set the property **PidLidReminderSet** on the **Exception Embedded Message Object** to **FALSE**, creating exceptions as appropriate (see [MS-OXOCAL] and [MS-OXCMSG] for details).

3.1.4.1.5 Setting Reminders on Draft Message Objects

By using the same mechanism that allows the client to set flagging properties for the sender of a **Message object** independently from the flagging properties that are sent to **recipients** [MS-OXOFLAG], a client can set the **reminder properties** for the sender of a Message object independently from the reminder properties that are sent to recipients.

If either a sender or recipient reminder is specified by a client that supports sender reminders, **PidTagSwappedToDoStore** MUST be set to enable post-transmit processing (see [MS-OXOFLAG] for details). The remaining work to process recipient and sender reminders is covered during post-transmit processing as specified in [MS-OXOFLAG].

3.1.4.1.5.1 Setting a Reminder for all Recipients

To set a **recipient reminder** on a **Draft Message object**, the client sets **PidLidReminderSet** to **TRUE** and sets **PidLidReminderTime**, **PidLidReminderSignalTime**, and **PidTagReplyTime** to the required signal time.

3.1.4.1.5.2 Setting a Reminder for the Sender

The client MUST set the field **rtmReminder** to the required signal time and the field **fReminderSet** to **TRUE** in the **PtypBinary** property **PidTagSwappedToDoData**. The client sets the corresponding validity bits (0x00000040 and 0x00000080) to 1 in the **dwFlags** field in **PidTagSwappedToDoData**. See [MS-OXOFLAG] for the **PidTagSwappedToDoData** property definition.

3.1.4.2 Post-Transmit Processing

The post-transmit processing required by the Reminder Settings protocol is identical to the post-transmit processing specified in [MS-OXOFLAG].

3.1.4.3 Post-Receive Processing

For received **Message objects** that are not time flagged objects [MS-OXOFLAG], if the property **PidTagReplyTime** exists, the value of **PidTagReplyTime** SHOULD<16> be copied into **PidLidReminderTime** and **PidLidReminderSignalTime**, and **PidLidReminderSet** SHOULD be set to **TRUE**.

3.1.4.4 Removing a Reminder

3.1.4.4.1 Single Instance Objects

The client sets **PidLidReminderSet** to **FALSE**, which disables the reminder.

If the object is neither a **Task object** nor a **Calendar object**, the client MAY<17> clear **PidLidReminderTime** and **PidLidReminderSignalTime** and SHOULD clear **PidTagReplyTime**.

If the object is a Calendar object, the client SHOULD<18> set **PidLidAutoStartCheck** to **FALSE**.

3.1.4.4.2 Recurring Calendar Objects

To remove the **reminder** for all instances, including any **exceptions**, the client MUST set **PidLidReminderSet** to **FALSE** on the **Recurring Calendar object**.

To remove the reminder for a single instance, the client ensures that an exception exists for the instance, set the value of the **ReminderSet** field for the corresponding exception in the recurrence **BLOB** to **FALSE**, and set the property **PidLidReminderSet** on the **Exception Embedded Message object** to FALSE, and SHOULD<19> set the property **PidLidAutoStartCheck** on the **Exception Embedded Message object** to **FALSE**. See [MS-OXOCAL] for details about how to modify the **appointment recurrence** BLOB (**PidLidAppointmentRecur**) and the **Exception Attachment object**, and see [MS-OXCMSG] for details about accessing the **Exception Embedded Message object**, which is stored in **PidTagAttachDataObject**.

3.1.4.4.3 Recurring Task Objects

To remove the **reminder** for all instances, the client sets **PidLidReminderSet** to **FALSE** on the recurring **Task object**, and SHOULD<20> ensure that **PidLidTaskResetReminder** is either not present or has the value **FALSE**.

Removing the reminder for a single instance of a **Recurring Task object** is not possible, because Recurring Task objects do not support **exceptions**. Therefore, the reminder can only be enabled or disabled for the entire recurrence pattern.

3.1.4.5 When a Reminder Becomes Overdue

3.1.4.5.1 Reminders That Are Ignored When They Become Overdue

Minimal domain clients SHOULD ignore reminders for the following types of objects, and **full domain** clients MUST ignore reminders for the following types of objects:

- Meeting-related objects - any object where the **PtypString** property **PidTagMessageClass** contains the prefix "IPM.Schedule." <21>
- Unsent mail - any object that contains the **PtypInteger32** property **PidTagMessageFlags** with the flag set corresponding to MSGFLAG_UNSENT (see [MS-OXCMSG] for details about message flags, and the definition of MSGFLAG_UNSENT)
- Objects that have the **PtypBinary** property **PidTagSwappedToDoStore** set. See [MS-OXOFLAG] for details.<22>.

3.1.4.5.2 Reminders That Are Auto-Dismissed When Overdue

Clients following the **full reminder domain** scope strategy, as specified in section 3.1.3.1.1, SHOULD<23> auto-**dismiss** any objects found outside the **minimal domain**, which also have a **signal time** (**PidLidReminderSignalTime**) that is earlier than the upgrade time

(**piReminderUpgradeTime**), so that the objects never have to be considered again. A reminder is auto-dismissed by setting **PidLidReminderSet** to **FALSE**.

3.1.4.5.3 Actions for Overdue Reminders That Are Not Ignored or Auto-Dismissed

When a **reminder** becomes overdue, the client commonly prompts the user to take action, such as **dismissing** or **snoozing**. To identify the reminder to the user (or user agent), the client can<24> use any properties, in addition to the following properties that the client SHOULD use to determine the **due time** for displaying how long until the object is due, or how much time has passed since the object became due.

When DueIn time is displayed, it is calculated from the Due time, as summarized in the following table:

Case	Due time
For non-Calendar objects (tasks and flagged objects<25>)	If the property **PidLidTaskDueDate** exists, the client SHOULD <26> use "TaskDueDate + default End-of-Day time", otherwise use **PidLidReminderTime** as the due time.For more information about the End-of-Day time, see [MS-OXOCFG].
For single instance **Calendar objects**	**PidLidReminderTime**
For **Recurring Calendar objects**	The client SHOULD use the start time of the latest instance with enabled reminder for which the time defined by evaluating the expression (StartTime – ReminderDelta) is less than or equal to the time encoded in **PidLidReminderSignalTime** on the Recurring Calendar object. See [MS-OXOCAL]. NOTE: **StartTime** and **ReminderDelta** could be overridden by **exceptions**.

If **PidLidReminderOverride** is set to **FALSE**, or not set, the client SHOULD<27> play the default reminder sound.

If **PidLidReminderOverride** is set to **TRUE**, the client SHOULD<28> use the play sound and **PidLidReminderFileParameter** from the reminder properties of the given object.

If **PidLidReminderPlaySound** is **TRUE**, the client SHOULD<29> use **PidLidReminderFileParameter** to locate the specified sound file, and then play it.

If **PidLidReminderPlaySound** is **FALSE**, the client SHOULD NOT<30> play a reminder sound.

If the client uses **PidLidReminderFileParameter** for purposes of playing a sound, and it does not represent a full path, the client can search for a matching file name.

3.1.4.5.3.1 Starting a Conference

If **PidLidConferencingCheck** and **PidLidAutoStartCheck** are both set to **TRUE**, the client can launch the conference. The client MAY launch a conference (if **PidLidConferencingType** is confNetMeeting), and use the properties **PidLidConferencingCheck**, **PidLidConferencingType**, **PidLidOrganizerAlias**, **PidLidDirectory**, **PidTagConversationTopic**, **PidLidCollaborateDoc**, **PidLidNetShowUrl**, and **PidLidAppointmentStateFlags**. For details, see [MS-OXOCAL].

3.1.4.6 Dismissing a Reminder

3.1.4.6.1 Dismissing for Single Instance Objects

For all **single instance objects**, the client MUST set **PidLidReminderSet** to **FALSE**.

In addition, for **single instance task objects**, the client can follow the guidance in the following section for **dismissing** a **Recurring Task object**.

3.1.4.6.2 Dismissing for Recurring Calendar Objects

If there is a future instance (including **exceptions**) with a pending **reminder** (in other words, not disabled individually on all future instances), the client MUST set **PidLidReminderSignalTime** on the **Recurring Calendar object** based on **PidLidReminderDelta** and the start of that instance (that is, NextInstanceStartTime - **PidLidReminderDelta**). If no more instances (including exceptions) have a pending reminder, it is recommended that the client avoid setting **PidLidReminderSet** to **FALSE**, and the client MUST set **PidLidReminderSignalTime** to the **PtypTime** value "Midnight (**UTC**) January 1, 4501" (value Low:0xA3DD4000 High:0x0CB34557).

It is recommended that the client avoid setting **PidLidReminderSet** to **FALSE** when dismissing **reminders** for **Recurring Calendar Objects**, even when no more instances require a reminder to signal. This is to preserve the user's intent to signal reminders, in case the **recurrence** is extended at a later date, to include instances in the future.

NOTE: Instances never "turn into" exceptions as a result of **dismissing**/**snoozing**.

3.1.4.6.3 Dismissing for Recurring Task Objects

The client MUST set **PidLidReminderSet** to **FALSE**, and SHOULD<31> set **PidLidTaskResetReminder** to **TRUE** to **dismiss** the **reminder**.

For task objects, if **PidLidReminderTime** is in the future, the client SHOULD<32> set **PidLidReminderSignalTime** to the same value as **PidLidReminderTime** so that the reminder will be signaled at the appropriate time for the next instance.

3.1.4.7 Snoozing a Reminder

3.1.4.7.1 Snoozing for Single Instance Objects

The client MUST set **PidLidReminderSignalTime** to the time specified by the user (or user agent).

3.1.4.7.2 Snoozing for Recurring Objects

The client MUST set **PidLidReminderSignalTime** to either the specified time, or the signal time for the next instance with an **active reminder** (for example, NextInstanceStart - ReminderDelta), whichever is earlier.

NOTE: Instances never "turn into" exceptions as a result of **dismissing/snoozing**.

3.1.4.8 Generating Instances for Recurring Task Objects

In addition to the details related to the semantics and the use of **PidLidTaskResetReminder** specified in [MS-OXOTASK], the following applies to generating instances for **Recurring Task objects**:

When a Recurring Task object is updated after one instance of the task is marked complete, if the **PtypBoolean** property **PidLidTaskResetReminder** is **TRUE**, or **PidLidReminderSet** is **TRUE**, a **reminder** SHOULD<33> be set on the Recurring Task object by setting both **PidLidReminderTime** and **PidLidReminderSignalTime** to the next **signal time**. These values SHOULD correspond to the same time as the previous reminder, but on a different day, such that the difference between the new signal time and the new task due date is the same as the difference between the old signal time and old task due date values. The client adjusts for Daylight Saving Time when necessary (if a change in DST occurs between the current and next instance), so the **reminder** signals at the same time of day.

After an instance of a recurring task that has an **active** reminder is complete, the future instance MUST have **PidLidReminderSet** set to **TRUE**, and **PidLidTaskResetReminder** SHOULD NOT<34> be **TRUE**. The now-completed instance MUST have **PidLidReminderSet** set to **FALSE**, and **PidLidTaskResetReminder** SHOULD NOT<35> be **TRUE**.

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

Before adding, removing, or modifying a **reminder** on any **Message objects**, the client has to ask the server to perform a mapping from named properties to property identifiers by using **RopGetPropertyIDsOfNames**, as follows:

Property	Property Set GUID	Name or ID
PidLidReminderSet	{00062008-0000-0000-C000-000000000046}	0x8503
PidLidReminderTime	{00062008-0000-0000-C000-000000000046}	0x8502
PidLidReminderDelta	{00062008-0000-0000-C000-000000000046}	0x8501
PidLidReminderSignalTime	{00062008-0000-0000-C000-000000000046}	0x8560
PidLidTaskDueDate	{00062003-0000-0000-C000-000000000046}	0x8105
PidLidTaskResetReminder	{00062003-0000-0000-C000-000000000046}	0x8107
PidLidAppointmentRecur	{00062002-0000-0000-C000-000000000046}	0x8216
PidLidAutoStartCheck	{00062002-0000-0000-C000-000000000046}	0x8244
PidLidFExceptionalAttendees	{00062002-0000-0000-C000-000000000046}	0x822B

The server might respond 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
PidLidReminderSet	0x8004

PidLidReminderTime	0x8005
PidLidReminderDelta	0x81FF
PidLidReminderSignalTime	0x8006
PidLidTaskDueDate	0x8144
PidLidTaskResetReminder	0x815B
PidLidAppointmentRecur	0x81AE
PidLidAutoStartCheck	0x82E0
PidLidFExceptionalAttendees	0x82D7

4.1 Set a Reminder on a Single Instance Appointment

John has an existing **appointment** for "Dinner with Robin Counts" at Coho Vineyard on February 15, 2008 from 6:00 P.M. to 7:00 P.M. Pacific Standard Time. Around 11:44 A.M., John sets a **reminder** on the appointment for 30 minutes.

Before John modified the object, some of the properties were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	44 00 69 00 6E 00 6E 00 65 00 72 00 20 00 77 00 69 00 74 00 68 00 20 00 52 00 6F 00 62 00 69 00 6E 00 20 00 43 00 6F 00 75 00 6E 00 74 00 73 00 00 00	"Dinner with Robin Counts"
PidLidLocation	0x8009	**PtypString**	43 00 6F 00 68 00 6F 00 20 00 56 00 69 00 6E 00 65 00 79 00 61 00 72 00 64 00 00 00	"Coho Vineyard"
PidTagStartDate	0x0060	**PtypTime**	00 10 00 A3 3F 70 C8 01	2008/02/16 02:00:00.000
PidTagEndDate	0x0061	**PtypTime**	00 78 C4 04 48 70 C8 01	2008/02/16 03:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	00	**FALSE**

Property Name	Property Identifier	Property Type	Data	Value
PidLidReminderTime	0x8005	**PtypTime**	00 10 00 A3 3F 70 C8 01	2008/02/16 02:00:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 F6 8E 8A 3D 70 C8 01	2008/02/16 01:45:00.000
PidLidReminderDelta	0x81FF	**PtypInteger32**	0F 00 00 00	15 minutes

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 then sends a **RopSetProperties** request with the following properties in response to user-entered data:

Property	Property Identifier	Property Type	Data	Value
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidLidReminderDelta	0x81FF	**PtypInteger32**	1E 00 00 00	30 minutes
PidLidReminderSignalTime	0x8006	**PtypTime**	00 DC 1D 72 3B 70 C8 01	2008/02/16 01:30:00.000

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

4.2 Set a Reminder on a Message Object

John, who is currently working in the Pacific Standard time zone, has an existing **message** titled "Important Project Information" that was previously flagged for follow-up by February 14, 2008. To ensure that it is not forgotten, John sets a reminder for 6:00 P.M.

Before John modified the object, some of its properties were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	49 00 6D 00 70 00 6F 00 72 00 74 00 61 00 6E 00 74 00 20 00 50 00 72 00 6F 00 6A 00 65 00	"Important Project Information"

Property Name	Property Identifier	Property Type	Data	Value
			63 00 74 00 20 00 49 00 6E 00 66 00 6F 00 72 00 6D 00 61 00 74 00 69 00 6F 00 6E 00 00 00	
PidLidTaskStartDate	0x8143	**PtypTime**	00 C0 A3 8A 9C 6E C8 01	2008/02/14 00:00:00.000
PidLidTaskDueDate	0x8144	**PtypTime**	00 C0 A3 8A 9C 6E C8 01	2008/02/14 00:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	00	**FALSE**
PidLidReminderTime	0x8005	**PtypTime**	Cleared	N/A
PidLidReminderSignalTime	0x8006	**PtypTime**	Cleared	N/A

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 then sends a **RopSetProperties** request with the following properties in response to user-entered data:

Property	Property Identifier	Property Type	Data	Value
PidLidReminderTime	0x8005	**PtypTime**	00 50 96 78 76 6f c8 01	2008/02/15 02:00:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 50 96 78 76 6f c8 01	2008/02/15 02:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidTagReplyTime	0x0030	**PtypTime**	00 50 96 78 76 6f c8 01	2008/02/15 02:00:00.000
PidTagReplyRequested	0x0C17	**PtypBoolean**	01	**TRUE**
PidTagResponseRequested	0x0063	**PtypBoolean**	01	**TRUE**
PidTagFlagStatus	0x1090	**PtypInteger32**	02 00 00 00	follow-up marked

Next, the client sends a **RopDeletePropertiesNoReplicate** request for the following properties:

Property	Property Identifier	Property Type
PidTagFlagCompleteTime	0x1091	**PtypTime**

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

4.3 Dismiss a Reminder on a Task

John had previously created a Task object titled "Prepare for Contoso presentation", with a **reminder** that was signaled on February 15, 2008, at 11:30 A.M. Pacific Standard Time. Because he is done, John **dismisses** the reminder at 11:31 A.M.

Before John dismissed the reminder, some of its properties were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	50 00 72 00 65 00 70 00 61 00 72 00 65 00 20 00 66 00 6F 00 72 00 20 00 43 00 6F 00 6E 00 74 00 6F 00 73 00 6F 00 20 00 70 00 72 00 65 00 73 00 65 00 6E 00 74 00 61 00 74 00 69 00 6F 00 6E 00 00 00	"Prepare for Contoso presentation"
PidLidTaskStartDate	0x8143	**PtypTime**	00 80 0D B5 65 6F C8 01	2008/02/15 00:00:00.000
PidLidTaskDueDate	0x8144	**PtypTime**	00 80 0D B5 65 6F C8 01	2008/02/15 00:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidLidReminderTime	0x8005	**PtypTime**	00 6C 83 27 09 70 C8 01	2008/02/15 19:30:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 6C 83 27 09 70 C8 01	2008/02/15 19:30:00.000

Property Name	Property Identifier	Property Type	Data	Value
PidLidTaskResetReminder	0x815B	**PtypBoolean**	Cleared	N/A

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 then sends a **RopSetProperties** request with the following properties in response to user-entered data:

Property	Property Identifier	Property Type	Data	Value
PidLidReminderSet	0x8004	**PtypBoolean**	00	**FALSE**
PidLidTaskResetReminder	0x815B	**PtypBoolean**	01	**TRUE**

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

4.4 Dismiss a Reminder on a Recurring Calendar Object

John has a recurring **appointment** for lunch with Ben Smith every Friday at noon Pacific Standard Time, with a 20 minute reminder. The reminder for the first instance is displayed, and John **dismisses** the reminder before leaving for the Coho Winery.

Before John dismissed the object, some of its properties were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	4C 00 75 00 6E 00 63 00 68 00 20 00 77 00 69 00 74 00 68 00 20 00 42 00 65 00 6E 00 20 00 53 00 6D 00 69 00 74 00 68 00 00 00	"Lunch with Ben Smith"
PidLidLocation	0x8009	**PtypString**	43 00 6F 00 68 00 6F 00 20 00 57 00 69 00 6E 00 65 00	"Coho Winery"

Property Name	Property Identifier	Property Type	Data	Value
			72 00 79 00 00 00	
PidTagStartDate	0x0060	**PtypTime**	00 A0 65 58 0D 70 C8 01	2008/02/15 20:00:00.000
PidTagEndDate	0x0061	**PtypTime**	00 08 2A BA 15 70 C8 01	2008/02/15 21:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidLidReminderTime	0x8005	**PtypTime**	00 A0 65 58 0D 70 C8 01	2008/02/15 20:00:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 28 24 8D 0A 70 C8 01	2008/02/15 19:40:00.000
PidLidReminderDelta	0x81FF	**PtypInteger32**	14 00 00 00	20 minutes
PidLidAppointmentRecur	0x81AE	**PtypBinary**	Cb: 50 00 Lpb: See below	Cb: 80 bytes Lpb: See [MS-OXOCAL]

The value of **PidLidAppointmentRecur** is as follows:

Cb: 50 00

Lpb:

```
0x0000: 04 30 04 30 0B 20 01 00-00 00 C0 21 00 00 01 00
0x0010: 00 00 00 00 00 00 20 00-00 00 23 20 00 00 0A 00
0x0020: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 40 4A
0x0030: C3 0C DF 80 E9 5A 06 30-00 00 09 30 00 00 D0 02
0x0040: 00 00 0C 03 00 00 00 00-00 00 00 00 00 00 00 00
```

See [MS-OXOCAL] for details about interpreting the data.

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 then sends a **RopSetProperties** request with the following properties in response to user-entered data:

Property	Property Identifier	Property Type	Data	Value
PidLidReminderSignalTime	0x8006	**PtypTime**	00 68 08 B6 8A 75 C8 01	2008/02/22 19:40:00.000

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

4.5 Snooze a Reminder on a Contact Object

John has an existing **Contact object** for Adam Barr with a reminder set so he is reminded to call on February 15, 2008 at 11:15 A.M., Pacific Standard Time. The **reminder** is displayed, but John is still in the middle of a project, so around 11:18 A.M., John **snoozes** the reminder for 1 hour.

Before John snoozed the object, some of its properties were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	41 00 64 00 61 00 6D 00 20 00 42 00 61 00 72 00 72 00 00 00	"Adam Barr"
PidLidTaskStartDate	0x8143	**PtypTime**	00 80 0D B5 65 6F C8 01	2008/02/15 00:00:00.000
PidLidTaskDueDate	0x8144	**PtypTime**	00 80 0D B5 65 6F C8 01	2008/02/15 00:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidLidReminderTime	0x8005	**PtypTime**	00 52 12 0F 07 70 C8 01	2008/02/15 19:15:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 52 12 0F 07 70 C8 01	2008/02/15 19:15:00.000

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 then sends a **RopSetProperties** request with the following properties in response to user-entered data:

Property	Property Identifier	Property Type	Data	Value
PidLidReminderSignalTime	0x8006	**PtypTime**	00 8C 20 DC 0F 70 C8 01	2008/02/15 20:18:00.000

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

4.6 Remove a Reminder From an Instance of a Recurring Calendar Object

John has an existing **recurring appointment** for lunch on Fridays at noon, Pacific Standard Time, but the instance for next week, which occurs on February 22, 2008, was previously changed to 11:00 A.M. (that is, the next instance already has an exception). John opens that instance, sets the **signal time** to none, and saves the object, disabling the **reminder** for just that instance.

Before John modified the object, some of the properties on the **Recurring Calendar object** were as follows:

Property Name	Property Identifier	Property Type	Data	Value
PidTagNormalizedSubject	0x0E1D	**PtypString**	4C 00 75 00 6E 00 63 00 68 00 20 00 77 00 69 00 74 00 68 00 20 00 42 00 65 00 6E 00 20 00 53 00 6D 00 69 00 74 00 68 00 00 00	"Lunch with Ben Smith"
PidTagStartDate	0x0060	**PtypTime**	00 A0 65 58 0D 70 C8 01	2008/02/15 20:00:00.000
PidTagEndDate	0x0061	**PtypTime**	00 08 2A BA 15 70 C8 01	2008/02/15 21:00:00.000
PidLidReminderSet	0x8004	**PtypBoolean**	01	**TRUE**
PidLidRemin	0x8005	**PtypTime**	00 A0 65 58 0D 70	2008/02/15

Property Name	Property Identifier	Property Type	Data	Value
derTime			C8 01	20:00:00.000
PidLidReminderSignalTime	0x8006	**PtypTime**	00 00 44 54 82 75 C8 01	2008/02/22 06:40:00.000
PidLidAppointmentRecur	0x81AE	**PtypBinary**	Cb: 72 00 Lpb: See below	Cb: 114 bytes Lpb: See [MS-OXOCAL]

The value of **PidLidAppointmentRecur** is as follows:

Cb: 72 00

Lpb:

```
0x0000: 04 30 04 30 0B 20 01 00-00 00 C0 21 00 00 01 00
0x0010: 00 00 00 00 00 00 20 00-00 00 23 20 00 00 0A 00
0x0020: 00 00 00 00 00 00 01 00-00 00 A0 71 C3 0C 01 00
0x0030: 00 00 A0 71 C3 0C 40 4A-C3 0C DF 80 E9 5A 06 30
0x0040: 00 00 09 30 00 00 D0 02-00 00 0C 03 00 00 01 00
0x0050: 34 74 C3 0C 70 74 C3 0C-70 74 C3 0C 00 00 00 00
0x0060: 00 00 04 00 00 00 00 00-00 00 00 00 00 00 00 00
0x0070: 00 00
```

See [MS-OXOCAL] for details about interpreting the data.

Some properties from the **Exception Attachment Object** are listed in the following table.

Property Name	Property Identifier	Property Type	Data	Value
PidTagAttachSize	0x0E20	PtypInteger32	DC 0E 00 00	3804 bytes
PidTagLastModificationTime	0x3008	PtypTime	62 BC D2 86 20 70 C8 01	2008/02/15 22:17:18.328

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

Then the client sends a **RopOpenAttachment** request and waits for the server to respond. The server response contains a handle to the **Exception Attachment object**. See [MS-OXOCAL] for details about using the **attachment** table to find the attachment that corresponds to a given **exception**.

Then the client sends a **RopOpenEmbeddedMessage** request and waits for the server to respond. The server response contains a handle to the **Exception Embedded Message object**.

Next, the client sends a **RopSetProperties** request with the following properties in response to the user-entered data for the Exception Embedded Message object (followed by a **RopSaveChangesMessage** request to persist the object on the server):

Property Name	Property Identifier	Property Type	Data	Value
PidLidAutoStartCheck	0x82E0	**PtypBoolean**	00	**FALSE**
PidLidReminderSet	0x8004	**PtypBoolean**	00	**FALSE**

Next, the client sends a **RopSetProperties** request with the following properties in response to user-entered data for the **Exception Attachment object** (followed by a **RopSaveChangesAttachment** request to persist the object on the server):

Property Name	Property Identifier	Property Type	Data	Value
PidTagAttachMethod	0x3705	**PtypInteger32**	05 00 00 00	5
PidTagRenderingPosition	0x370B	**PtypInteger32**	FF FF FF FF	-1
PidTagExceptionStartTime	0x7FFB	**PtypTime**	00 38 62 11 42 75 c8 01	2008/02/22 11:00:00.000
PidTagAttachEncoding	0x3702	**PtypBinary**	00 00	Size: 0 bytes

Property Name	Property Identifier	Property Type	Data	Value
PidTagExceptionEndTime	0x7FFC	**PtypTime**	00 a0 26 73 4a 75 c8 01	2008/02/22 12:00:00.000
PidTagAttachmentFlags	0x7FFD	**PtypInteger32**	02 00 00 00	Exception to a recurrence
PidTagDisplayName	0x3001	**PtypString**	55 00 6e 00 74 00 69 00 74 00 6c 00 65 00 64 00 00 00	"Untitled"
PidTagAttachmentLinkId	0x7FFA	**PtypInteger32**	00 00 00 00	0
PidTagAttachFlags	0x3714	**PtypInteger32**	00 00 00 00	0
PidTagAttachmentHidden	0x7FFE	**PtypBoolean**	01	**TRUE**
PidTagAttachmentContactPhoto	0x7FFF	**PtypBoolean**	00	**FALSE**

Next, the client sends a **RopSetProperties** request with the following properties in response to user-entered data for the Recurring Calendar object (followed by a **RopSaveChangesMessage** request to persist the object on the server):

Property Name	Property Identifier	Property Type	Data	Value
PidLidAppointmentRecur	0x81AE	**PtypBinary**	Cb: 76 00 Lpb: See below	Cb: 118 bytes Lpb: See below
PidLidFExceptionalAttendees	0x82D7	**PtypBoolean**	00	**FALSE**
PidLidReminderSignalTime	0x8006	**PtypTime**	00 A8 EC DE 0A 7B C8 01	2008/02/29 19:40:00.000

Then the client sends a **RopRelease** request to release the **Exception Embedded Message object**, and a **RopRelease** request to release the **Exception Attachment object**.

Finally, the client sends a **RopRelease** request to release the **attachment** table, and a **RopRelease** request to release the Recurring Calendar object.

The value of **PidLidAppointmentRecur** is as follows:

Cb: 76 00

Lpb:

```
0x0000: 04 30 04 30 0B 20 01 00-00 00 C0 21 00 00 01 00
0x0010: 00 00 00 00 00 00 20 00-00 00 23 20 00 00 0A 00
0x0020: 00 00 00 00 00 00 01 00-00 00 A0 71 C3 0C 01 00
0x0030: 00 00 A0 71 C3 0C 40 4A-C3 0C DF 80 E9 5A 06 30
0x0040: 00 00 09 30 00 00 D0 02-00 00 0C 03 00 00 01 00
0x0050: 34 74 C3 0C 70 74 C3 0C-70 74 C3 0C 08 00 00 00
0x0060: 00 00 00 00 00 00 04 00-00 00 00 00 00 00 00 00
0x0070: 00 00 00 00 00 00
```

This corresponds to the following table, in which the **OverrideFlags** and **fReminder** fields for the first exception are the only two fields that changed within the **PtypBinary** property **PidLidAppointmentRecur**. In this specific case, the 2 byte value, starting at the 93rd byte (Flags), was changed from 0x0000 to 0x0008, and a 4 byte value representing **FALSE** (0x00000000) was inserted between the 94th and 95th bytes, extending the size of the property from 114 bytes to 118 bytes. See [MS-OXOCAL] for details about how to determine the byte position in other cases.

Name	Type	Size	Data	Description
ReaderVersion	**WORD**	2	`04 30`	This field is set to 0x3004.
WriterVersion	**WORD**	2	`04 30`	This field is set to 0x3004.
UIGroup	**WORD**	2	`0b 20`	The pattern of the recurrence is weekly (0x200b).
PatternType	**WORD**	2	`01 00`	The pattern type is weekly (0x0001).
CalendarType	**WORD**	2	`00 00`	The calendar type is Gregorian (0x0000).

Name	Type	Size	Data	Description
FirstDateTime	**ULONG**	4	c0 21 00 00	See [MS-OXOCAL] for details about how this property was calculated based on the StartDate.
Period	**ULONG**	4	01 00 00 00	The recurrence occurs every week (0x00000001).
SlidingFlag	**ULONG**	4	00 00 00 00	The recurring instances do not rely on completion of the previous instances.
PatternTypeSpecific	**Byte Array**	Varies (4 bytes, in this case, for Weekly pattern)	20 00 00 00	The recurring appointment occurs on Friday. See [MS-OXOCAL] for details about how the value is determined.
EndType	**ULONG**	4	23 20 00 00	No End. (0x00002023)
OccurrenceCount	**ULONG**	4	0A 00 00 00	Ignored, because recurrence does not have an end.
FirstDOW	**ULONG**	4	00 00 00 00	The first day of the week on the calendar is Sunday (the default value).
DeletedInstanceCount	**ULONG**	4	01 00 00 00	One deleted instance.
Deleted Instance Start	**ULONG**	4	A0 71 C3 0C	0x0CC371A0 == February 22, 2008 at 00:00 in UTC.
ModifiedInstanceCount	**ULONG**	4	01 00 00 00	One modified instance.

Name	Type	Size	Data	Description
Modified Instance Start	**ULONG**	4	A0 71 C3 0C	0x0CC371A0 == February 22, 2008 at 00:00 in UTC.
StartDate of entire series	**ULONG**	4	40 4A C3 0C	The start date of the recurrence given in minutes since midnight January 1, 1601 corresponds to February 15, 2008 12:00:00 A.M. 0x0CC34A40
EndDate of entire series	**ULONG**	4	DF 80 E9 5A	The end date of the recurrence given in minutes since midnight January 1, 1601, however, since there is no end date, this value corresponds to the "last valid" date: December 31, 4500 11:59 P.M. (0x5AE980DF)
ReaderVersion2	**ULONG**	4	06 30 00 00	This value is set to 0x00003006.
WriterVersion2	**ULONG**	4	09 30 00 00	This value is set to 0x00003009.<36>
StartTimeOffset	**ULONG**	4	D0 02 00 00	The hexadecimal start time of the recurrence is 0x000002D0, which corresponds to 720 in decimal. 720 minutes is 12 hours, which is 12 P.M.
EndTimeOffset	**ULONG**	4	0C 03 00 00	The hexadecimal end time of the recurrence is 0x00000030C, which corresponds to 780 minutes, which is 1:00 P.M.
ExceptionCount	**WORD**	2	01 00	There is one exception in this recurrence **BLOB**.
ExceptionInfo block for exception 1:				
Start Date Time	**ULONG**	4	34 74 C3 0C	0x0CC37434 == February 22, 2008 at 11:00 A.M.
End Date Time	**ULONG**	4	70 74 C3 0C	0x0CC37470 == February 22, 2008 at 12:00 P.M.
Original Start Time	**ULONG**	4	70 74 C3 0C	0x0CC37470 == February 22, 2008 at 12:00 P.M.
OverrideFlags	**WORD**	2	08 00	0x0008 corresponds to the "fReminderSet" flag, meaning just that field is specified for this exception.

Name	Type	Size	Data	Description
fReminderSet	**ULONG**	4	00 00 00 00	0x00000000 == FALSE, meaning ReminderSet is overridden with the value FALSE for this exception.
ReservedBlock1Size	**ULONG**	4	00 00 00 00	There is no data in the reserved block.
ExtendedException block for exception 1:				
ChangeHighlight	**Byte Array**	Varies	04 00 00 00 00 00 00 00	The size of the ChangeHighlight is 4. The value of the **PidLidChangeHighlight** property is zero for this exception.
ReservedBlockEE1Size	**ULONG**	4	00 00 00 00	There is no data in the reserved block.
ReservedBlock2Size	**ULONG**	4	00 00 00 00	There is no data in the reserved block.

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to the Reminder Settings protocol. General security considerations pertaining to the underlying transport apply, as specified in [MS-OXCMSG] and [MS-OXCPRPT].

5.2 Index of Security Parameters

None.

6 Appendix A: Office/Exchange Behavior

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

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

<1> Section 2.2.1.5: Exchange Server user interfaces do not honor or set the property **PidLidReminderOverride**.

<2> Section 2.2.1.6: Exchange Server user interfaces do not honor or set the property **PidLidReminderPlaySound**.

<3> Section 2.2.1.7: Outlook supports whatever the Windows API WINMM!PlaySound is able to play.

<4> Section 2.2.1.7: The Exchange Server user interfaces do not honor or set the property **PidLidReminderFileParameter**.

<5> Section 2.2.1.9: Outlook 2003, Outlook 2007, and Outlook 2010 preserves the value if already set.

<6> Section 2.2.2.3.1: Outlook 2003 does not set **PidLidTaskDueDate** for non-Task objects, so when a reminder signals for a flagable object, the "due in" time is calculated based on **PidLidReminderTime**. Outlook 2007 SP1 does set **PidLidTaskDueDate** for flaggable objects, so has a more intuitive "due in" time when the reminder signals. Outlook does not set **PidLidTaskDueDate** for tasks, when the user only specifies a **reminder** on a **Task object** without also specifying a **due date**. In other words, this property can be set on a Task object in the context of setting a due date. The task due date is independent of the **signal time**. See [MS-OXOTASK] for complete details about the conditions where **PidLidTaskDueDate** is set on a Task object.

<7> Section 2.2.2.4: Exchange Server user interfaces do not honor or set the property **PidLidAutoStartCheck**.

<8> Section 2.2.2.4: Exchange Server user interfaces do not honor or set the property **PidLidConferencingCheck**.

<9> Section 3.1.3.1.1: Outlook 2003 follows the **minimal reminder domain** scope strategy. Exchange Server 2003 signals **reminders** only from the Calendar and Tasks folders.

<10> Section 3.1.3.1.1: Using multiple clients that use different **reminder** domains can lead to inconsistent reminder signaling behavior. For example, such an environment might have different overdue reminders signaled in different clients, which can lead to unexpectedly missed reminders when using the client with the smaller reminder domain.

<11> Section 3.1.3.1.1.1: Not supported in Outlook 2003. Outlook 2007 and Outlook 2010 sets **piReminderUpgradeTime** to 0, when the "cleanreminders" feature is invoked, via command-line parameter, to clean the **reminders search folder**, after which all reminders would be eligible to signal again. Under normal operation, Outlook does not change **piReminderUpgradeTime** after the value is initialized; it does this only when the "cleanreminders" command-line option is used with Outlook 2007 and Outlook 2010.

<12> Section 3.1.4.1.2: Outlook 2003, Outlook 2007, Outlook 2010, Exchange Server 2007, and Exchange Server 2010 user interface set the property **PidTagReplyTime** when setting a **reminder** on an already-flagged object. Setting **PidTagReplyTime** is not required for reminders to signal locally, but this property is used to transmit a reminder on a sent **message**. The Exchange Server 2003 user interface does not expose a way to set reminders on objects that are neither **Calendar objects** nor **Task objects**, so this constraint does not apply.

<13> Section 3.1.4.1.2: Outlook does not set all the specified properties at "reminder creation" time, because some of the properties were set in a previous operation (such as object creation or flagging for follow-up), or are set independent of the reminder itself (such as for **Task objects** and **PidLidTaskDueDate**), and the values did not change during the reminder creation process. Because Outlook does not expose UI to just set a reminder on an e-mail message without also flagging, some properties such as **PidLidRequest** happen to get set by Outlook when a reminder is added to an e-mail message. These properties are not required for the reminder to function, but are listed for completeness.

<14> Section 3.1.4.1.3: The Exchange Server 2007 and Exchange Server 2010 user **interfaces sets the property PidLidReminderTime to the start time for the next** instance with an active reminder that has an end date in the future. This is true regardless of whether the **Recurring Calendar object** is created where part of the instances occur in the past, or when **dismissing** a **reminder**. When a reminder is dismissed (or a recurrence is created where part of the recurrence is in the past), **PidLidReminderTime** is updated to match the start time of the next instance with an **active reminder**. Outlook 2003, Outlook 2007, Outlook 2010, Exchange Server 2003 user interface set **PidLidReminderTime** to the start time of the first instance, and do not modify the **PidLidReminderTime** property during **snooze** or dismiss operations after the property is set on a Recurring Calendar object, even if the value has been modified. To clarify, Outlook 2003, Outlook 2007, Outlook 2010, and the Exchange Server 2003 user interface only set the **PidLidReminderTime** property when creating the Recurring Calendar object or modifying the series such that the series start date changes, and otherwise do not modify the property. In other words, the property **PidLidReminderTime** cannot be assumed to have a specific value for recurring objects.

<15> Section 3.1.4.1.3: The Exchange Server 2007 and Exchange Server 2010 user interface sets the property **PidLidReminderTime** to the start time for the next instance with an active **reminder** that has an end date in the future. This is true regardless of whether the **Recurring Calendar object** is created where part of the instances occur in the past, or when **dismissing** a

reminder. When a reminder is dismissed (or a recurrence is created where part of the recurrence is in the past), **PidLidReminderTime** is updated to match the start time of the next instance with an **active reminder**. Outlook 2003, Outlook 2007, Outlook 2010, Exchange Server 2003 user interface set **PidLidReminderTime** to the start time of the first instance, and do not modify the **PidLidReminderTime** property during **snooze** or dismiss operations after the property is set on a Recurring Calendar object, even if the value has been modified. To clarify, Outlook 2003, Outlook 2007, Outlook 2010,and the Exchange Server 2003 user interface only set the **PidLidReminderTime** property when creating the Recurring Calendar object or modifying the series such that the series start date changes, and otherwise do not modify the property. In other words, the property **PidLidReminderTime** cannot be assumed to have a specific value for recurring objects.

<16> Section 3.1.4.3: Exchange Server user interface does not perform this processing.

<17> Section 3.1.4.4.1: Outlook 2003 also deletes **PidTagReplyTime** for an object that is neither a **Calendar object** nor a **Task object**. Outlook 2007 deletes the properties **PidLidReminderTime** and **PidLidReminderSignalTime** and **PidTagReplyTime** when removing the reminder from an object that is neither a Calendar object nor a Task object. For the remaining cases, Outlook 2003 and Outlook 2007 do not delete the properties **PidLidReminderTime** or **PidLidReminderSignalTime** when removing the reminder for Calendar or Task objects; this is for user convenience to persist the **signal time** values even when the **reminder** is disabled.

<18> Section 3.1.4.4.1: The Exchange Server user interfaces do not honor or set the property **PidLidAutoStartCheck**.

<19> Section 3.1.4.4.2: The Exchange Server user interfaces do not honor or set the property **PidLidAutoStartCheck**.

<20> Section 3.1.4.4.3: The Exchange user interfaces do not honor or set **PidLidTaskResetReminder**.

<21> Section 3.1.4.5.1: Exchange 2007 and Exchange 2010 user interface uses stricter criteria when ignoring meeting-related objects, by ignoring messages with **PidTagMessageClass** prefixed with "IPM.Schedule.Meeting". Outlook 2007 Outlook 2010 uses slightly more lenient criteria: "IPM.Schedule" (note the missing trailing dot).

<22> Section 3.1.4.5.1: Outlook 2003 does not ignore objects that have **PidTagSwappedTodoStore** set. Outlook 2003 also does not ignore meeting-related objects, but this is not a problem because it uses the **minimal reminder domain**, which does not include **special folders** like Sent Items.

<23> Section 3.1.4.5.2: Exchange Server 2003, Exchange Server 2007, and Exchange Server 2010 never read the property **piReminderUpgradeTime**. Exchange Server 2003 does not auto-**dismiss reminders** that appear outside the **minimal reminder domain**. Exchange

Server 2007 and Exchange Server 2009 does auto-dismiss said reminders, but only on initial transition to using the **full reminder domain** on a given mailbox store.

<24> Section 3.1.4.5.3: Outlook 2003, Outlook 2007, and Outlook 2010 use the properties **PidTagNormalizedSubject**, **PidLidRequest**, and **PidLidLocation**. Outlook 2007 and Outlook 2010 uses **PidLidTaskDueDate**.

<25> Section 3.1.4.5.3: Outlook 2003 displays DueIn "Now" when first signaling a **reminder** for a task or flagged object,. Outlook 2007 and Outlook 2010 displays a more intuitive time-until-DueDate value for these object types. Outlook 2007 and Outlook 2010 uses the property **PidLidTaskDueDate** and the "end of workday" values to determine the **due time**. Outlook 2003, Exchange Server 2003 user interface use the property **PidLidReminderTime** as the due time.

<26> Section 3.1.4.5.3: Outlook 2003 uses the **signal time** as the "due time".

<27> Section 3.1.4.5.3: The Exchange Server user interfaces do not use **PidLidReminderOverride**, **PidLidReminderPlaySound**, or **PidLidReminderFileParameter**.

<28> Section 3.1.4.5.3: The Exchange Server user interfaces do not use **PidLidReminderOverride**, **PidLidReminderPlaySound**, or **PidLidReminderFileParameter**.

<29> Section 3.1.4.5.3: The Exchange Server user interfaces do not use **PidLidReminderOverride**, **PidLidReminderPlaySound**, or **PidLidReminderFileParameter**.

<30> Section 3.1.4.5.3: The Exchange Server user interfaces do not use **PidLidReminderOverride**, **PidLidReminderPlaySound**, or **PidLidReminderFileParameter**.

<31> Section 3.1.4.6.3: Exchange Server User Interfaces do not set **PidLidTaskResetReminder** when **dismissing** a **reminder** on a **Task object.** This means that Exchange does not support dismissing reminders on single instances of **Recurring Task Objects**. Outlook does set **PidLidTaskResetReminder**.

<32> Section 3.1.4.6.3: Exchange Server 2003 user interface sets **PidLidReminderSignalTime** when **dismissing Task objects**.

<33> Section 3.1.4.8: Exchange Server 2003 user interfaces sets these properties if **PidLidReminderSet** is **TRUE**.

<34> Section 3.1.4.8: Exchange Server user interfaces do not honor or set **PidLidTaskResetReminder**.

<35> Section 3.1.4.8: Exchange Server user interfaces do not honor or set **PidLidTaskResetReminder**.

<36 > Section 4.6: Outlook 2003 uses a different format for this property stream and sets the WriterVersion2 property to 0x00003008..

Index