

[MS-OXOPFFB]: Public Folder–Based Free/Busy 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	Updated references to reflect date of initial release.
Microsoft Corporation	September 3, 2008	1.02	Updated references.
Microsoft Corporation	December 3, 2008	1.03	Revised and edited technical content.
Microsoft Corporation	April 10, 2009	2.0	Updated technical content and applicable product releases.

Table of Contents

1	Introduction.....	5
1.1	Glossary.....	5
1.2	References.....	7
1.2.1	Normative References.....	7
1.2.2	Informative References.....	8
1.3	Protocol Overview.....	8
1.4	Relationship to Other Protocols.....	8
1.5	Prerequisites/Preconditions.....	8
1.6	Applicability Statement.....	8
1.7	Versioning and Capability Negotiation.....	8
1.8	Vendor-Extensible Fields.....	9
1.9	Standards Assignments.....	9
2	Messages.....	9
2.1	Transport.....	9
2.2	Message Syntax.....	9
2.2.1	Free/Busy Message.....	9
2.2.1.1	Common Properties.....	9
2.2.1.1.1	PidTagNormalizedSubject.....	9
2.2.1.1.2	PidTagMessageClass.....	9
2.2.1.2	Free/Busy Properties.....	9
2.2.1.2.1	PidTagScheduleInfoMonthsTentative.....	9
2.2.1.2.2	PidTagScheduleInfoFreeBusyTentative.....	10
2.2.1.2.3	PidTagScheduleInfoMonthsBusy.....	10
2.2.1.2.4	PidTagScheduleInfoFreeBusyBusy.....	10
2.2.1.2.5	PidTagScheduleInfoMonthsAway.....	11
2.2.1.2.6	PidTagScheduleInfoFreeBusyAway.....	11
2.2.1.2.7	PidTagScheduleInfoMonthsMerged.....	11
2.2.1.2.8	PidTagScheduleInfoFreeBusyMerged.....	11
2.2.1.2.9	PidTagFreeBusyPublishStart.....	11
2.2.1.2.10	PidTagFreeBusyPublishEnd.....	11
2.2.1.2.11	PidTagFreeBusyRangeTimestamp.....	11
2.2.1.2.12	PidTagFreeBusyMessageEmailAddress.....	12
2.2.1.3	Delegate Information Properties.....	12
2.2.1.4	Deprecated Properties.....	12
2.2.1.4.1	PidTagGatewayNeedsToRefresh.....	12
2.2.1.4.2	PidTagScheduleInfoResourceType.....	12
2.2.1.4.3	PidTagScheduleInfoFreeBusy.....	12
2.2.2	Public Folder Free/Busy Related Properties.....	12
2.2.2.1	PidTagFreeBusyEntryIds.....	12
3	Protocol Details.....	13

3.1	Client Details	13
3.1.1	Abstract Data Model	13
3.1.1.1	Non-Persisted Free/Busy Related Properties	13
3.1.1.1.1	PidTagSchedulePlusFreeBusyEntryId	13
3.1.2	Timers	14
3.1.3	Initialization	14
3.1.4	Higher-Layer Triggered Events	14
3.1.4.1	Publishing Free/Busy Data.....	14
3.1.4.1.1	Determining the Data Set for Publishing.....	14
3.1.4.1.2	Finding the Free/Busy Message.....	15
3.1.5	Message Processing Events and Sequencing Rules.....	16
3.1.6	Timer Events.....	16
3.1.7	Other Local Events	17
4	Protocol Examples.....	17
4.1	Updating the Free/Busy Message.....	17
4.2	Finding Free/Busy Messages By Using E-mail Addresses.....	19
4.3	PidTagScheduleInfoMonthsBusy Calculation	32
4.4	PidTagScheduleInfoFreeBusyBusy Calculation	34
4.4.1	PidTagScheduleInfoFreeBusyBusy Calculation with Two Non-Consecutive Events	35
4.4.2	PidTagScheduleInfoFreeBusyBusy Calculation with Two Consecutive Events	36
4.4.3	PidTagScheduleInfoFreeBusyBusy Calculation with Events in Multiple Months	36
4.4.4	PidTagScheduleInfoFreeBusyBusy Calculation When an Event is Spread Across Multiple Months	36
4.5	PidTagScheduleInfoFreeBusyMerged Calculation.....	37
5	Security.....	37
5.1	Security Considerations for Implementers.....	37
5.2	Index of Security Parameters.....	37
6	Appendix A: Office/Exchange Behavior.....	37
	Index.....	40

1 Introduction

The Public Folder–Based Free/Busy protocol is a format that is used to publish information that describes the availability of an **attendee** or **resource**. <1> This information can be used by a broad range of consumers to efficiently schedule meetings and/or provide presence information.

This document specifies the following:

- The format in which **free/busy** data is represented.
- A method for publishing data in the prescribed format.
- A method for discovering and interpreting data in the prescribed format.

1.1 Glossary

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

attendee
Address Book object
ambiguous name resolution (ANR)
appointment
Calendar object
Coordinated Universal Time (UTC)
delegate
Delegate Information object
distinguished name (DN)
EntryID
folder
folder ID (FID)
free/busy
handle
little-endian
message
message ID (MID)
Message object
non-IPM subtree
Out of Office (OOF)
property
property ID
public folder
remote operation (ROP)
resource
special folder
store

The following data type is defined in [MS-DTYP]:

BYTE

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

PtypBinary
PtypBoolean
PtypInteger32
PtypMultipleBinary
PtypMultipleInteger32
PtypString
PtypTime

The following terms are specific to this document:

Availability service: A Web service that provides **free/busy** information.

Busy: One of the possible values of the **free/busy status** on an **appointment**. A **Busy** status indicates that the user is not available for other **appointments** during this time.

calendar: See **Calendar object.F**

conflict: A condition that is created when an **appointment** is scheduled at the same time as another appointment.

end of range: The end date of a **publishing range**.

Free: One of the possible values of the **free/busy status** on an **appointment**. A **Free** status indicates that the user is available during this **appointment**.

free/busy message: A message in the **public folder store** that contains **free/busy** data.

free/busy status: A status that indicates how an **appointment** on the **calendar** of an **attendee** or **resource** affects their availability. The **free/busy status** is the value of the **PidLidBusyStatus property** of the **appointment**, as specified in [MS-OXOCAL].

Tentative: One of the possible values of the **free/busy status** on an **appointment**. A **Tentative** status indicates that the user is tentatively booked during this **appointment**.

publishing: Writing **free/busy** data to a shared location.

publishing range: The number of months of **calendar** data to be published.

start of range: The start date of a **publishing range**.

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

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

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

[MS-OXDSCLI] Microsoft Corporation, "Autodiscover Publishing and Lookup Protocol Specification", June 2008.

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

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

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

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

[RFC1279] Hardcastle-Kille, S.E., "X.500 and Domains", RFC 1279, November 1991, <http://www.ietf.org/rfc/rfc1279.txt>.

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

1.2.2 Informative References

None.

1.3 Protocol Overview

Free/busy data is derived from **calendar** data and falls into four categories: **Free**, **Busy**, **Tentative** and **Out of Office (OOF)**.

The Public Folder–Based Free/Busy protocol specifies how free/busy data is read from and written to a shared location so that it can be used to effectively and efficiently schedule meetings.

Free/busy data can also be obtained by using the Availability Web service, as specified in [MS-OXWAVLS], and by directly accessing the calendar of a user and reading the free/busy status **property**, as specified in [MS-OXOCAL].

1.4 Relationship to Other Protocols

The Public Folder–Based Free/Busy protocol extends the Message and Attachment Object protocol [MS-OXCMSG].

1.5 Prerequisites/Preconditions

The Public Folder–Based Free/Busy protocol assumes that the server is configured to support **public folders**.<2>

1.6 Applicability Statement

The Public Folder–Based Free/Busy protocol is appropriate for use by higher layers of a server or client implementation that schedule meetings to avoid scheduling **conflicts**.

The following related functionality is best accomplished by using other protocols.

- Displaying complete **calendar** details. This is best accomplished by receiving permissions and directly browsing the calendar of the other **attendee** or **resource**. For more details, see [MS-OXODLGT].
- Displaying **free/busy** data outside the **publishing range**. This is best accomplished by using the **Availability service**, as specified in [MS-OXWAVLS].

It is recommended that this specification only be used if the Availability service is not supported by the server.<3> For more details about the Availability service, see [MS-OXDSCLI].

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The Public Folder–Based Free/Busy protocol uses the protocol specified in [MS-OXCMSG] as its primary transport mechanism.

2.2 Message Syntax

2.2.1 Free/Busy Message

Free/busy data is represented as a set of properties that are set on a **message** in the **public folders store**. The message is referred to as the **free/busy message**. The location of this message is specified in section 3. Unless otherwise specified, the free/busy message conforms to a **Message object**, as specified in [MS-OXCMSG].

2.2.1.1 Common Properties

2.2.1.1.1 PidTagNormalizedSubject

A **PtypString** property that specifies the subject of the **free/busy message**. Its value is a string that is derived from the e-mail address of the user. The e-mail address is the value of the **PidTagEmailAddress** property of the **Address Book object** of the user. The subject is derived by taking the sub-string, starting with "/cn", prepending "USER", and converting all the characters to uppercase.

2.2.1.1.2 PidTagMessageClass

A **PtypString** property that **MUST** be set to "IPM.Post".

2.2.1.2 Free/Busy Properties

2.2.1.2.1 PidTagScheduleInfoMonthsTentative

A **PtypMultipleInteger32** property that specifies the months for which **free/busy** data of type **Tentative** is present in the **free/busy message**. The number of **PtypInteger32** values in this property **MUST** be between 0 (zero) and the number of months covered by the **publishing range**, which is the period between **PidTagFreeBusyPublishStart** and **PidTagFreeBusyPublishEnd**.

Each value in this **PtypMultipleInteger32** property, interpreted as a signed **PtypInteger32**, has a month and year encoded in it. This is calculated by using the expression "year × 16 + month" where year and month are based on the Gregorian **calendar**. The values are sorted in ascending order and are encoded in **little-endian** format.

If an event is spread across multiple months, or multiple years, there **MUST** be one value for each of the months that fall in the publishing range. If there are no **Tentative** events in the publishing range, this property and **PidTagScheduleInfoFreeBusyTentative** are not set or, if they already exist, are deleted.

2.2.1.2.2 PidTagScheduleInfoFreeBusyTentative

A **PtypMultipleBinary** property that specifies the blocks of times for which the **free/busy status** is **Tentative**. This property has as many values as the number of values in **PidTagScheduleInfoMonthsTentative**. Each binary value represents a month and corresponds to the value at the same index in **PidTagScheduleInfoMonthsTentative**. The binary values are sorted in the same order as the values in **PidTagScheduleInfoMonthsTentative**.

Each binary value has one or more 4-BYTE blocks and each of them contains the start time in the first two bytes and end time in the second two bytes in **little-endian** format. The start time is the number of minutes between 12:00 A.M. **Coordinated Universal Time (UTC)** of the first day of the month and the start time of the event in UTC. The end time is the number of minutes between 12:00 A.M. UTC of the first day of the month and the end time of the event in UTC. The 4-BYTE blocks are sorted in ascending order.

Consecutive or overlapping blocks of time are merged into one block with start time as the start time of the first block and end time as the end time of the last block. If an event is spread across multiple months or years, the event is split into multiple blocks, one for each month. If there are no **Tentative** events in the **publishing range**, this **property** and **PidTagScheduleInfoMonthsTentative** are not set or, if they already exist, are deleted. Otherwise, this property **MUST** be set.

2.2.1.2.3 PidTagScheduleInfoMonthsBusy

A **PtypMultipleInteger32** property that specifies the months for which **free/busy** data of type **busy** is present in the **free/busy message**. The format, computation, and constraints of this property are the same as those for **PidTagScheduleInfoMonthsTentative** but refer to **appointments** that are marked **busy** on the associated **calendar**.

2.2.1.2.4 PidTagScheduleInfoFreeBusyBusy

A **PtypMultipleBinary** property that specifies the blocks of time for which the **free/busy status** is type **busy**. The format, computation, and constraints of this property are the same as those for **PidTagScheduleInfoFreeBusyTentative**, but refer to **appointments** that are marked **busy** on the associated **Calendar object**.

2.2.1.2.5 PidTagScheduleInfoMonthsAway

A **PtypMultipleInteger32** property that specifies the months for which **free/busy** data of type **OOF** is present in the **free/busy message**. The format, computation, and constraints of this property are the same as those for **PidTagScheduleInfoMonthsTentative**, but refer to **appointments** that are marked **OOF** on the associated **Calendar object**.

2.2.1.2.6 PidTagScheduleInfoFreeBusyAway

A **PtypMultipleBinary** property that specifies the times for which the **free/busy status** is set to **OOF**. The format, computation, and constraints of this property are the same as those for **PidTagScheduleInfoFreeBusyTentative** but refer to **appointments** that are marked **OOF** on the associated **Calendar object**.

2.2.1.2.7 PidTagScheduleInfoMonthsMerged

A **PtypMultipleInteger32** property that specifies the months for which **free/busy** data of type **busy** or **OOF** is present in the **free/busy message**. Events of **free/busy** type **Tentative** are not included in this property. The syntax/format and constraints of this property are the same as those for **PidTagScheduleInfoMonthsTentative** but refer to **appointments** that are marked **OOF** or **Busy** on the associated **Calendar object**.

2.2.1.2.8 PidTagScheduleInfoFreeBusyMerged

A **PtypMultipleBinary** property that specifies the times for which the **free/busy status** is set to **Busy** or **OOF**. Events of **free/busy** type **Tentative** are not included in this property. The format, computation, and the restrictions of this property are the same as those for **PidTagScheduleInfoFreeBusyTentative**, but refer to **appointments** that are marked **OOF** or **Busy** on the associated **Calendar object**.

2.2.1.2.9 PidTagFreeBusyPublishStart

A signed **PtypInteger32** property that specifies the start time of the **publishing range**. This value is expressed as the number of minutes since midnight, January 1, 1601 in **UTC**.

2.2.1.2.10 PidTagFreeBusyPublishEnd

A signed **PtypInteger32** property that specifies the end time of the **publishing range**. It is computed by adding the value of **PidTagFreeBusyCountMonths** to the start date of the publishing range. This value is expressed as the number of minutes since midnight, January 1, 1601 in **UTC**.

2.2.1.2.11 PidTagFreeBusyRangeTimestamp

A **PtypTime** property that specifies the time that the data was published, in **UTC**.

2.2.1.2.12 PidTagFreeBusyMessageEmailAddress

A **PtypString** property that specifies the e-mail address of the user to whom this **free/busy message** applies. The value of this property is the same as the **PidTagEmailAddress** property value of the **Address Book object** for the **attendee** or **resource**.

2.2.1.3 Delegate Information Properties

The following **properties** are used in the Delegate Access Configuration protocol [MS-OXODLGT] and the Appointment and Meeting Object protocol [MS-OXOCAL]:

- **PidTagScheduleInfoAutoAcceptAppointments**
- **PidTagScheduleInfoDisallowRecurringAppts**
- **PidTagScheduleInfoDisallowOverlappingAppts**
- **PidTagScheduleInfoDelegatorWantsCopy**
- **PidTagScheduleInfoDontMailDelegates**
- **PidTagScheduleInfoDelegatorWantsInfo**
- **PidTagFreeBusyCountMonths**

These properties are optional on a **free/busy message**. If they are set on the **free/busy message**, the values of these properties **MUST** be equal to the value of the same property on the **Delegate Information object**, as specified in [MS-OXOCAL].

2.2.1.4 Deprecated Properties

2.2.1.4.1 PidTagGatewayNeedsToRefresh

A **PtypBoolean** property. This property is deprecated and **SHOULD NOT** be used.<4>

2.2.1.4.2 PidTagScheduleInfoResourceType

A signed **PtypInteger32** property that is set to 0 when sending and ignored on receipt.<5>

2.2.1.4.3 PidTagScheduleInfoFreeBusy

A **PtypBinary** property. This property is deprecated. It **SHOULD NOT** be set and **MUST** be ignored.<6>

2.2.2 Public Folder Free/Busy Related Properties

2.2.2.1 PidTagFreeBusyEntryIds

A **PtypMultipleBinary** property. This property is set on the root **folder** and on the **Inbox special folder** of the local **store**. The value on the root folder of the local store **MUST** be equal to the value on the **Inbox special folder**. The property value contains four binary values, set as follows:

- The first value is set to NULL.
- The second value is set to the **EntryID** of the delegate information object. For more details about the delegate information message, see [MS-OXODLGT].
- The third value can be set to the EntryID of the free/busy message of the logged on user. It is set to NULL if the **free/busy public folder** is unavailable. This value is set when a client or server creates the **free/busy message** for the first time for a user.
- The fourth value is set to the EntryID of the folder the PidTagDisplayName property value for which is equal to "FreeBusy Data", and which is a child folder of the root folder of the store.

3 Protocol Details

3.1 Client Details

Free/busy data is kept in a specific **message** in the **public folders store**. There is one message for each user for whom free/busy data is published. This message is called the **free/busy message**. It is contained in a **folder** that represents the administrative group to which the user belongs. There is one folder for each administrative group in the organization. All these folders are descendants of the **special folder** "SCHEDULE+ FREE BUSY". Each folder that represents an administrative group is a sibling of the folders that represent other administrative groups. An administrative group represents an "Organisational Unit" within an organization, as specified in [RFC1279]. It is the *org-unit-rdn* component of the **distinguished name (DN)** of the **Address Book object** for the user, as defined in [MS-OXOABK] section 2.2.1.1.

A client can read or write free/busy data for another user. For more details about finding the free/busy message for a specific user, see section 3.1.4.1.2.

3.1.1 Abstract Data Model

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

3.1.1.1 Non-Persisted Free/Busy Related Properties

Some **properties** are not persisted on any object but are temporarily cached in memory or computed on demand. The **free/busy**-related properties specified in this section are not persisted.

3.1.1.1.1 PidTagSchedulePlusFreeBusyEntryId

A **PtypBinary property** that contains the **EntryID** of the **folder** named "SCHEDULE+ FREE BUSY" under the NON_IPM_SUBTREE of the **public folder store**. This property is

not stored on any object but is computed on demand. The **folder ID (FID)** of "SCHEDULE+FREE BUSY" is returned in response to **RopLogon**. The FID is preserved and used to compute the EntryID when it is needed.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Publishing Free/Busy Data

The **free/busy** data is published whenever **appointments** are created or deleted or when any of the following three **properties** of an appointment are modified:

PidLidAppointmentStartWhole, **PidLidAppointmentEndWhole**, **PidLidBusyStatus**.

A client or server keeps a local copy of free/busy data and publishes the changes periodically or immediately after the data changes.

The client performs the following steps to publish the data:

1. Determine the data to be published by using the steps specified in section 3.1.4.1.1.
2. Find the **free/busy message** by using the steps specified in section 3.1.4.1.2.
3. If the **message** does not exist, create a new message. Use step 2 to determine the **subfolder** and the subject of the message.
4. If the message was created, set the third binary value of the **PidTagFreeBusyEntryIds** property to the **EntryID** of the free/busy message. For more information about how to create a message, see [MS-OXCMSG].

3.1.4.1.1 Determining the Data Set for Publishing

Data to be published is determined by finding all the **calendar** events that are in the **publishing range**. The publishing range is calculated as follows:

- **Start of range** is 12:00 A.M. UTC on the first day of the month or the first day of the week, whichever occurs earlier at the time of **publishing**.
- **End of range** is calculated by adding the value of **PidTagFreeBusyCountMonths** to start of range.
- All the **appointments** with an end time that is greater than the start of range and start time less than the end of range are considered to be in the publishing range.

3.1.4.1.2 Finding the Free/Busy Message

The **free/busy messages** are stored in a descendant **folder** of the "SCHEDULE+ FREE BUSY" folder, under the NON_IPM_SUBTREE of the **public folders store**. There **MUST** be only one free/busy message for each user. A client can read or write **free/busy** data for users other than the logged on users. The client **MUST** find the free/busy message that corresponds to the user before reading or writing information for that user. At the time of **publishing**, the client determines whether a free/busy message already exists for the user in question before creating one.

To locate the free/busy message that corresponds to a given user, do one of the following:

- To find the free/busy message of the logged on user, get the **EntryID** of the free/busy message. The **EntryID** of the free/busy message is set in the third binary value of the **PidTagFreeBusyEntryIds** property for the Inbox. Use the EntryID of the free/busy message to generate the **message ID** of the message, as specified in section 2.2.1.3.1 of [MS-OXCDATA]. Use the message ID of the message to open the **message**, as specified in [MS-OXCMSG].
- To find the free/busy message of another user, use the name (full name or part of their name) and perform an **ambiguous name resolution (ANR)** to get their **Address Book object**, as specified in [MS-OXOABK] section 3.1.4.4.

If the **PidTagFreeBusyEntryIds** property does not exist or if the third binary value of this property is empty, or if the free/busy message is being requested for another user, the subject and the name of the sub-folder can be used to find the free/busy message, as specified in sections 3.1.4.1.2.1 through 3.1.4.1.2.4.

3.1.4.1.2.1 Determining the E-mail Address

To determine the e-mail address, get the **Address Book object** and then do one of the following:

- If the client is **publishing** the data for the logged on user, find the Address Book object of the logged on user by matching its **EntryID** with the **PidTagMailboxOwnerEntryId** of the **store**.
- If the client is publishing the data for another user, get the Address Book object by performing an **ANR** by using the full name or partial name of the delegator. For details about how to resolve a name, see [MS-OXOABK] section 3.1.4.4.
- Get the e-mail address from the **PidTagEmailAddress** property of the Address Book object.

3.1.4.1.2.2 Determining the Name of the Appropriate Sub-Folder of the "SCHEDULE+ FREE BUSY" Folder

To determine the name of the appropriate sub-**folder**, perform the following steps:

1. Derive the name of the sub-folder by truncating the e-mail address up to, but not including, the first occurrence of "/cn".
2. Prepend the string "EX:" to the value computed in step 1.
3. Find the folder by using the name derived in the step 1. For details about how to find a folder by name, see [MS-OXCFOLD].

3.1.4.1.2.3 Determining the Subject of the Appropriate Message

To determine the subject of the appropriate message, perform the following steps:

1. Derive the subject by taking the sub-string that starts with the first occurrence of "/cn" in the e-mail address.
2. Prepend the string "USER-" to the value computed in step 1.
3. Convert all the characters to upper-case.

3.1.4.1.2.4 Finding the Message

To find the message, perform the following steps:

1. Derive the **folder** name and the subject of the **message** by using the logic specified in section 3.1.4.1.2.2 and 3.1.4.1.2.3.
2. Open the "SCHEDULE + FREE BUSY" folder by using **PidTagSchedulePlusFreeBusyEntryId** property.
3. Open the specific folder for this user by using the name derived by using the steps in specified section 3.1.4.1.2.2.
4. Find the specific **message** for this user by finding the message in which **PidTagSubject** equals the subject derived by using the steps specified in section 3.1.4.1.2.3. For more details about how to find a message that matches a subject, see [MS-OXCTABL] section 2.2.2.5.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

4.1 Updating the Free/Busy Message

The following example describes how the **free/busy message** of a user is updated by the client after the user adds new **appointments** to their **calendar**.

Joe sets his **publishing** interval to three months.

Joe creates the following appointments on his calendar. All the events have a **free/busy status** type of **busy**.

Note: All times in this example are in Pacific Time.

Feb 2nd

Appointment time: 12:00 P.M. – 1:00 P.M.

Appointment time: 1:00 P.M. – 2:00 P.M.

April 2nd

Appointment time: 12:00 P.M. – 1:00 P.M.

Appointment time: 3:00 P.M. – 4: P.M.

The client publishes the availability information by using **RopCreateMessage** to create the **message** in the sub-**folder** for the administrative group of the user under the "SCHEDULE + FREE/BUSY" folder. The server returns a success code and a **handle** to a **Message object**.

Then the client uses **RopSetProperties** to transmit availability data to the server. The properties listed in the following table are sent.

Property	Property ID	Property Type	Value
PidTagScheduleInfoResourceType	0x6841	Signed PtypInteger32	0x00000000
PidTagGatewayNeedsToRef	0x684	PtypBoolean	0x00000001

resh	6		
PidTagNormalizedSubject	0x0E1D	PtypString	USER- /CN=RECIPIENTS/CN=JOE
PidTagScheduleInfoMonthsMerged	0x684F	Signed PtypMultipleInteger32	See section 4.3. PidTagScheduleInfoMonthsMerged is calculated similarly to PidTagScheduleInfoMonthsBusy .
PidTagScheduleInfoFreeBusyMerged	0x6850	PtypMultipleBinary	See section 4.5.
PidTagScheduleInfoMonthsBusy	0x6853	Signed PtypMultipleInteger32	See section 4.3.
PidTagScheduleInfoFreeBusyBusy	0x6854	Signed PtypMultipleBinary	See section 4.4.
PidTagFreeBusyPublishStart	0x6847	Signed PtypInteger32	214105440
PidTagFreeBusyPublishEnd	0x6848	Signed PtypInteger32	214234980
PidTagFreeBusyRangeTimeStamp	0x6868	PtypTime	2008/02/22 01:13:00.000

Because there are no events with a free/busy type of **Tentative** or **OOF**, the client does not set **PidTagScheduleInfoFreeBusyTentative**, **PidTagScheduleInfoMonthsTentative**, **PidTagScheduleInfoFreeBusyAway**, or **PidTagScheduleInfoMonthsAway**.

The client then uses **RopSaveChanges** to commit the properties on the server, and then **RopRelease** to release the Message object.

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

4.2 Finding Free/Busy Messages By Using E-mail Addresses

The following example describes how to find the **free/busy message** of a user by using their e-mail address.

First, determine the **folder** name in which the free/busy message exists and determine the subject by using the e-mail address of the user.

For example, a user who has a **PidTagEmailAddress** value of "o=Adventure-Works/ou=New York/cn= Recipients/cn=David" would have his or her **free/busy** data stored in the folder named "EX:/o=Adventure-Works/ou=New York" and in a **message** with subject equal to "USER-/CN=RECIPIENTS/CN=David".

For details about how to get the **Address Book object** of a user and the value of the **PidTagEmailAddress** property, see [MS-OXOABK].

The folder name and subject have now been determined.

The following is an example of the **ROPs** the client uses to find the free/busy message. David has one event in his **calendar** that starts on December 25th, 2007 at 7:10 P.M. and ends on Dec 25th, 2008 at 7:10 P.M., Pacific Time.

RopLogon Request <7>

RopId: 0xFE

LogonId: 0

OutputHandleIndex: 0 (HSOT=0xffffffff)

LogonFlags: 0x04 Replicated

OpenFlags: 0x00000406 PUBLIC HOME_LOGON NO_MAIL

StoreState: 0x00000000 <none>

EssdnSize: 0x00

Essdn <no mailbox>

Response to **RopLogon**

RopLogon

RopId: 0xFE

OutputHandleIndex: 0 (HSOT=0x00000017)

ReturnValue: ecNone (success) (0x00000000)

LogonFlags: 0x04 Replicated

Public store

FolderArray:

FolderID 1: 0001-000000000006 Root Folder

FolderID 2: 0001-000000000001 IPM subtree

FolderID 3: 0001-000000000002 Non-IPM subtree

FolderID 4: 0001-000000000003 Eforms registry

FolderID 5: 0001-000000000004

FolderID 6: 0001-000000000005 Offline address book

FolderID 7: 0000-000000000000 Local Eforms registry

FolderID 8: 0003-000000000007 SCHEDULE+ FREEBUSY

FolderID 9: 0004-000000000008 Local address book

FolderID 10: 0000-000000000000 Article index

FolderID 11: 0000-000000000000 Schedule

FolderID 12: 0000-000000000000 <not used>

FolderID 13: 0000-000000000000 <not used>

ServerGUID: a608eae8-6603-4509-89b3-6dac886dca4d

PublicFolderPerUserGUID: fbdd61f1-4863-4d07-902a-64d07f4ca88d

Open the folder named SCHEDULE+ FREEBUSY, which has an ID of 0001-000000000004 as determined from the **RopLogon** response.

RopOpenFolder <8> Request

RopId: 0x02

LogonId: 5

InputHandleIndex: 0 (HSOT=0x0000008c)

FolderId: 0001-000000000004

OpenModeFlags: 0x00 ReadOnly

The response to **RopOpenFolder** is omitted for readability and clarity.

To find the sub-folder that matches the name that was determined earlier, the client sends the following ROP requests:

RopGetHierarchyTable Request

RopId: 0x04

LogonId: 5

InputHandleIndex: 0 (HSOT=0x0000008b)

OutputHandleIndex: 1 (HSOT=0xffffffff)

TableFlags: 0x00 Standard

The response to **RopGetHierarchyTable** has been omitted for readability and clarity.

RopSetColumns Request

RopId: 0x12

LogonId: 5

InputHandleIndex: 1 (HSOT=0xffffffff)

SetColumnsFlag: 0x00 Wait

PropertyTagCount: 3 (0x03)

PropertyTags: 0x67480014 **PidTagFolderId**

0x3001001F **PidTagDisplayName**

The response to **RopSetColumns** has been omitted for readability and clarity.

RopFindRow Request <9>

RopId: 0x4F

LogonId: 5

InputHandleIndex: 1 (HSOT=0xffffffff)

FindRowFlags: 0x00 Direction: forward

RestrictionDataSize: 0x0078 (120)

RestrictionData:

ConditionType: 0x04 RES_PROPERTY:

RelationalOperator: 0x04 RELOP_EQ

0x3001001F **PidTagDisplayName** EX:/o=Adventure-Works/ou=New

York

Origin: 0x00 BOOKMARK_BEGINNING

BookmarkSize: 0x0000 (0)

Response to **RopFindRow**

RopId: 0x4F

InputHandleIndex: 1 (HSOT=0x00000039)

ReturnValue: ecNone (success) (0x00000000)

RowNoLongerVisible: 0x00 (FALSE)

HasRowData: 0x01 (TRUE)

RowData:

HasError: 0

PropertyArray:

PropCount: 3

0x67480014 **PidTagFolderId** 0x070000000000000003

0x3001001F **PidTagDisplayName** EX:/o=Adventure-Works/ou=New York

Now that the folder has been found, and the **FID** was retrieved by using **RopFindRow**, open the folder.

RopOpenFolder Request

RopId: 0x02

LogonId: 5

InputHandleIndex: 0 (HSOT=0x0000008c)

FolderId: 0003-000000000007

OpenModeFlags: 0x00 ReadOnly

The response to **RopOpenFolder** has been omitted for readability and clarity.

RopGetContentsTable Request

RopId: 0x05

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000023)

OutputHandleTable: 1 (HSOT=0xffffffff)

TableFlags: 0x00 Standard

RopSetColumns Request

RopId: 0x12

LogonId: 0

InputHandleIndex: 1 (HSOT=0xffffffff)

SetColumnFlags: 0x00 Wait

PropertyTagCount: 5 (0x05)

PropertyTags: 0x67480014 **PidTagFolderId**

0x674A0014 **PidTagMid**

0x674D0014 **PidTagInstID**

0x674E0003 **PidTagInstanceNum**

0x0E1D001F **PidTagNormalizedSubject**

RopSortTable Request

RopId: 0x13

LogonId: 0

InputHandleIndex: 1 (HSOT=0xffffffff)

SortTableFlags: 0x00 Wait

SortOrderCount: 0x0001 (1)

CategoryCount: 0x0000 (0)

ExpandedCount: 0x0000 (0)

SortOrders:

PropertyTag: 0x0E1D001F **PidTagNormalizedSubject**

0x00 Flag: TABLE_SORT_ASCEND

RopFindRow Request

RopId: 0x4F

LogonId: 0

InputHandleIndex: 1 (HSOT=0xffffffff)

FindRowFlags: 0x00 Direction: forward

RestrictionDataSize: 0x0054 (84)

RestrictionData:

ConditionType: 0x04 RES_PROPERTY:

RelationalOperator: 0x04 RELOP_EQ

0x0E1D001F **PidTagNormalizedSubject** USER-
/CN=RECIPIENTS/CN=DAVID

Origin: 0x00 BOOKMARK_BEGINNING

BookmarkSize: 0x0000 (0)

Response to **RopGetContentsTable**

RopId: 0x05

OutputHandleIndex: 1 (HSOT=0x00000022)

ReturnValue: ecNone (success) (0x00000000)

RowCount: 113

Response to **RopSetColumns**

RopId: 0x12

InputHandleIndex: 1 (HSOT=0x00000022)

ReturnValue: ecNone (success) (0x00000000)

TableStatus: TBLSTAT_COMPLETE (0x00)

Response to **RopSortTable**

RopId: 0x13

InputHandleIndex: 1 (HSOT=0x00000022)

ReturnValue: ecNone (success) (0x00000000)

TableStatus: TBLSTAT_COMPLETE (0x00)

Response to **RopFindRow**

RopId: 0x4F

InputHandleIndex: 1 (HSOT=0x00000022)

ReturnValue: NotFound (0x8004010f)

The message is not found, so create it:

RopCreateMessage Request

RopId: 0x06

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000023)

OutputHandleIndex: 1 (HSOT=0xffffffff)

CodePageId: 0x0FFF (4095) <10>

FolderId: 0003-000000000007

AssociatedFlag: 0x00

Response to **RopCreateMessage**

RopId: 0x06

OutputHandleIndex: 1 (HSOT=0x00000020)

ReturnValue: ecNone (success) (0x00000000)

HasMessageId: 0

Now set the properties:

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x000F (15)

PropertyValueCount: 2 (0x02)

PropertyValues: 0x68410003 **PidTagScheduleInfoResourceType**
0x00000000 (0)

0x6846000B **PidTagGatewayNeedsToRefresh** 0x0001 (TRUE)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x0156 (342)

PropertyValueCount: 7 (0x07)

PropertyValues: 0x68410003 **PidTagScheduleInfoResourceType**

(TRUE) 0x6842000B **PidTagScheduleInfoDelegatorWantsCopy** 0x0001

0x6843000B **PidTagScheduleInfoDontMailDelegates** 0x0001 (TRUE)

(FALSE) 0x686D000B **PidTagScheduleInfoAutoAcceptAppointments** 0x0000

(FALSE) 0x686E000B **PidTagScheduleInfoDisallowRecurringAppts** 0x0000

(FALSE) 0x686F000B **PidTagScheduleInfoDisallowOverlappingAppts** 0x0000

0x684B000B **PidTagScheduleInfoDelegatorWantsInfo** 0x0001 (TRUE)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x0056 (86)

PropertyValueCount: 2 (0x02)

PropertyValues: 0x003D001F **PidTagSubjectPrefix** (null)

0x0E1D001F **PidTagNormalizedSubject** USER-
/CN=RECIPIENTS/CN=DAVID

RopSaveChangesMessage Request

RopId: 0x0C

LogonId: 0

ResponseHandleIndex: 1 (HSOT=0x00000023)

InputHandleIndex: 0 (HSOT=0x00000020)

SaveFlags: 0x0A KeepOpenReadWrite DelayedCall

Response to **RopSaveChangesMessage:**

RopSaveChangesMessage

RopId: 0x0C

ResponseHandleIndex: 1 (HSOT=0x00000023)

ReturnValue: ecNone (success) (0x00000000)

InputHandleIndex: 0 (HSOT=0x00000020)

MessageID: 0001-0000000051e3

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 2 (HSOT=0x00000020)

PropertyValueSize: 0x00AC (172)

PropertyValueCount: 1 (0x01)

PropertyValues: 0x6849001F **PidTagFreeBusyMessageEmailAddress**
/o=Adventure-Works/ou=New York/cn=Recipients/cn=David

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x0026 (38)

PropertyValueCount: 2 (0x02)

PropertyValues: 0x684F1003 **PidTagScheduleInfoMonthsMerged**

PtypMultipleInteger32[0]: 32130

PtypMultipleInteger32[1]: 32131

0x68501102 **PidTagScheduleInfoFreeBusyMerged**

PtypMultipleBinary[0] (4 bytes):

0000: E0 01 20 A3

PtypMultipleBinary[1] (4 bytes):

0000: 00 00 E0 01

RopDeletePropertiesNoReplicate Request

RopId: 0x7A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyTagCount: 2 (0x02)

PropertyTags: 0x68511003 **PidTagScheduleInfoMonthsTentative**

0x68521102 **PidTagScheduleInfoFreeBusyTentative**

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x0026 (38)

PropertyValueCount: 2 (0x02)

PropertyValues: 0x68531003 **PidTagScheduleInfoMonthsBusy**

PtypMultipleInteger32[0]: 32130

PtypMultipleInteger32[1]: 32131

0x68541102 **PidTagScheduleInfoFreeBusyBusy**

PtypMultipleBinary [0] (4 bytes):

0000: E0 01 20 A3

PtypMultipleBinary [1] (4 bytes):

0000: 00 00 E0 01

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x0007 (7)

PropertyValueCount: 1 (0x01)

PropertyValues: 0x6846000B **PidTagGatewayNeedsToRefresh** 0x0001

(TRUE)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x000A (10)

PropertyValueCount: 1 (0x01)

PropertyValues: 0x68470003 **PidTagFreeBusyPublishStart**
PtypInteger32 0x0CC2FD60 (214105440)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x000A (10)

PropertyValueCount: 1 (0x01)

PropertyValues: 0x68480003 **PidTagFreeBusyPublishEnd** 0x0CC3A080
(214147200)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

InputHandleIndex: 0 (HSOT=0x00000020)

PropertyValueSize: 0x000E (14)

PropertyValueCount: 1 (0x01)

PropertyValues: 0x68680040 **PidTagFreeBusyRangeTimestamp**

High: 0x01C87A68

Low: 0x430A6000 (2008/02/29 00:16:00.000)

RopSetProperties Request

RopId: 0x0A

LogonId: 0

(0) **InputHandleIndex:** 0 (HSOT=0x00000020)
PropertyValueSize: 0x000A (10)
PropertyValueCount: 1 (0x01)
PropertyValues: 0x68410003 **PidTagScheduleInfoResourceType** 0x00000000

RopSaveChangesMessage Request

RopId: 0x0C
LogonId: 0
ResponseHandleIndex: 1 (HSOT=0x00000023)
InputHandleIndex: 0 (HSOT=0x00000020)
SaveFlags: 0x08 DelayedCall

Response to **RopSaveChangesMessage**:

RopSaveChangesMessage
RopId: 0x0C
ResponseHandleIndex: 1 (HSOT=0x00000023)
ReturnValue: ecNone (success) (0x00000000)
InputHandleIndex: 0 (HSOT=0x00000020)
MessageID: 0001-0000000051e3

The client then calls **ROPRelease** on all open folders and the newly created message.

4.3 *PidTagScheduleInfoMonthsBusy Calculation*

The **PidTagScheduleInfoMonthsBusy** property is calculated by using the following equations:

1. $(Year \times 16) + month$
2. *Convert result of equation 1 to hexadecimal*

Consider the following example:

Free/busy range is three months from the time of **publishing**. The time of publishing is 12:00 A.M. on Feb 25th UTC. There is at least one **calendar** item with a free/busy type of **Busy** in the months of February, March, and April.

Therefore, data in the months of February, March, April, and May will be published. More specifically:

- Feb 25th 12:00 A.M. to Feb 29th 11:59 P.M.
- March 1st 12:00 A.M. to March 31st 11:59 P.M.
- April 1st 12:00 A.M. to April 30th 11:59 P.M.
- May 1st 12:00 A.M. to May 25th 12: A.M.

All the times are in UTC. The property value is determined by using the following calculations.

For February
Year = 2008 Month = February = 2 $(2008 * 16) + 2 = 32130$ 32130 converted to hexadecimal = 7d82

For March
Year = 2008 Month = March = 3 $(2008 * 16) + 3 = 32131$ 32131 converted to hexadecimal = 7d83

For April
Year = 2008 Month = April = 4 $(2008 * 16) + 4 = 32132$ 32132 converted to hexadecimal = 7d84

The **PtypMultipleInteger32** will have three values with 0x00007D82, 0x00007D83, 0x00007D84 in each of the LONG values. Because the month of May did not have any events, there will not be a value that represents May. Otherwise, there would have been four values.

If the month of March, for example, did not have any **calendar** items with a free/busy type of **Busy**, there would be only two values.

4.4 *PidTagScheduleInfoFreeBusyBusy Calculation*

The **PidTagScheduleInfoFreeBusyBusy** property is calculated by using the following equations:

- Start time is the number of minutes between midnight on the first day of the month and the start time of the event in hexadecimal.

$$\text{Number of days before the scheduled date in that particular month} \times 24 \text{ (hours/day)} * 60 \text{ (minutes/hour)} + \text{(hour of start time on a 24-hour clock adjusted for UTC)} * 60 \text{ (minutes/hour)} = \text{Text value of start time in minutes}$$

Note: Each time zone requires a different adjustment for UTC.
- End time is the number of minutes between midnight on the first day of the month and the end time of the event in hexadecimal.

For example, assume there is only one event with a **free/busy status** of **Busy** that occurs during the months of February, March, April, and May. It occurs between noon and 1:00 P.M. Pacific Time on February 2nd. The **publishing** interval is three months. The time of publishing is 12:00 A.M. on Feb 25th UTC.

The value is determined by using the following calculations.

Start time
<p><i>Number of days before the scheduled date in that particular month = 1</i></p> <p><i>Start hour = noon = 12 on a 24-hour clock + 8 for UTC conversion = 20</i></p> <p>Note:</p> <p>+8 is the difference between UTC and Pacific Time. The adjustment is different for each time zone.</p> <p><i>Start time in minutes = (1 * 60 * 24) + (20 *</i></p>

Start time

$$60) = 1440 + 1200 = 2640$$

Hexadecimal start time = 0A50

End time

Number of days before the scheduled date in that particular month = 1

$$\text{End hour} = 1:00 \text{ P.M. Pacific Time} = 13 \text{ on a 24-hour clock} + 8 \text{ for UTC conversion} = 21$$

Note:

+8 is the difference between UTC and Pacific Time. The adjustment is different for each time zone.

$$\text{End time in minutes} = (1 * 60 * 24) + (21 * 60) = 1440 + 1260 = 2700$$

Hexadecimal end time = 0A8C

Note: In some cases, the UTC conversion changes the date of the start time or end time.

The multi-value binary will have one binary value. The binary value will have the following four bytes, encoded in **little-endian** — the first two bytes for the start time and the second two bytes for the end time: 0x50, 0x0A, 0x8C, 0x50.

4.4.1 PidTagScheduleInfoFreeBusyBusy Calculation with Two Non-Consecutive Events

Assume that there are two events with a **free/busy status** of **busy**, in the month of February.

February 2nd (times are in Pacific Time)

Appointment time: 12:00 P.M. to 1:00 P.M.

Appointment time: 3:00 P.M. to 4:00 P.M.

The **PtypMultipleBinary** will have one binary value. The binary value will have the following eight bytes: 0x50, 0x0A, 0x8C, 0x0A, 0x04, 0x0B, 0x40, 0x0B.

4.4.2 PidTagScheduleInfoFreeBusyBusy Calculation with Two Consecutive Events

If there are two consecutive events with the same **free/busy status**, the two times are merged together. For example, assume the following two events are of type **Busy**.

February 2nd (times are in Pacific Time)

Appointment time: 12:00 P.M. to 1:00 P.M.

Appointment time: 1:00 P.M. to 2:00 P.M.

There will be only one block with a start time of 12:00 P.M. and an end time of 2:00 P.M. The **PtypMultipleBinary** will have one binary value. The binary value will have the following four bytes: 0x50, 0x0A, 0xC8, 0x0A.

4.4.3 PidTagScheduleInfoFreeBusyBusy Calculation with Events in Multiple Months

Assume that there are two events of type **Busy** in the month of February and two events in the month of April. There are no events in March.

February 2nd (times are in Pacific Time)

Appointment time: 12:00 P.M. to 1:00 P.M.

Appointment time: 1:00 P.M. to 2:00 P.M.

April 2nd

Appointment time: 12:00 P.M. to 1:00 P.M.

Appointment time: 3:00 P.M. to 4:00 P.M.

The **PtypMultipleBinary** will have two binary values.

The first binary value will have the following four bytes: 0x50, 0x0A, 0xC8, 0x0A

The second binary value will have the following eight bytes: 0x14, 0x0A, 0x50, 0x0A, 0xC8, 0x0A, 0x04, 0x0B.

Note that the time difference between **UTC** and **PST** in April is +7 hours and therefore the values are calculated by adding 7 hours to the PST time.

4.4.4 PidTagScheduleInfoFreeBusyBusy Calculation When an Event is Spread Across Multiple Months

Consider an example where an event starts on Dec 25th, 2007, 9:00 A.M. and ends on Dec 25th, 2008, 10:00 A.M. This event is considered as a series of events that start and end as follows:

Dec 25th 2007, 9:00 A.M. — Dec 31st 2007, 12:00 A.M.

Jan 1st 2008, 12:00 A.M. — Jan 31st 2008, 12:00 A.M.

....

....

Dec 1st 2008, 12:00 A.M. — Dec 25th, 10:00 A.M.

The values are calculated as specified in the examples in sections 4.4.1 through 4.4.3.

If the **publishing** interval is 12 months and the time of publishing is 12:00 A.M. Dec 25th, 2007 UTC, there would be 13 binary values in both **PidTagScheduleInfoMonthsBusy** and **PidTagScheduleInfoFreeBusyBusy** properties; one for Dec 2007 and one for each month in 2008.

If the publishing interval is 12 months and the time of publishing is 12:00 A.M. Feb 25, 2007, UTC, there would be 11 binary values.

If the publishing interval is one month and the time of publishing is 12:00 A.M. Feb 25, 2007, UTC, there would be two binary values, for the months of Feb and March.

4.5 PidTagScheduleInfoFreeBusyMerged Calculation

The **PidTagScheduleInfoFreeBusyMerged** property includes events of type **busy** and **OOF**. For example, if a user has an event from 1:00 P.M. to 2:00 P.M. of type busy and another from 4:00 P.M. to 5:00 P.M. of type OOF, **PidTagScheduleInfoFreeBusyMerged** will contain two binary values.

If there are consecutive or overlapping scheduled events of type busy and OOF, they will be merged into one block. For example, if a user has an event from 1:00 P.M. to 2:00 P.M. of type busy, and another from 1:30 P.M. to 3:00 P.M. of type OOF, the **PidTagScheduleInfoFreeBusyMerged** property will contain one binary value with a start time of 1:00 P.M. and an end time of 3:00 P.M.

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to this protocol. General security considerations pertaining to the underlying protocol as specified in [MS-OXOMSG] apply.

5.2 Index of Security Parameters

None.

6 Appendix A: Office/Exchange Behavior

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

- Microsoft 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> Exchange 2010 Beta supports public folder referrals, but does not support public folders when client connection services are deployed on an Exchange server that does not also have a mailbox store installed.

<2> Section 1.5: Public folders are supported by default by Exchange Server 2003. **Public folders** might have to be configured explicitly on an Exchange 2007 server.

<3 > Section 1.6: Exchange 2003 does not support the Availability service.

<4> Section 2.2.1.4.1: Outlook 2003 and Outlook 2007 will set **PidTagGatewayNeedsToRefresh** to 1. This property is not read by Outlook 2003, Outlook 2007, Exchange 2003, or Exchange 2007.

<5> Section 2.2.1.4.2: The **PidTagSchdInfoResourceType** from the **free/busy message** is read from and written to by Outlook 2003 and Outlook 2007, but the value is not used.

<6> Section 2.2.1.4.3: The **PidTagSchdinfoFreebusy** property MAY be set on the **Delegate Info object** by versions of Outlook prior to Outlook 2003. This property is deprecated and is no longer read from or written to.

<7> Exchange 2010 Beta can output unexpected results when using **RopLogon** when client connection services are deployed on an Exchange server that does not also have a mailbox store installed.

<8> Exchange 2010 Beta can output unexpected results when using **RopOpenFolder** when client connection services are deployed on an Exchange server that does not also have a mailbox store installed.

<9> Exchange 2010 Beta does not support custom bookmarks when client connection services are deployed on an Exchange server that does not also have a mailbox store installed.

<10> Exchange 2010 Beta does not support non-English, non-United States locales or codepage when client connection services are deployed on an Exchange server that does not also have a mailbox store installed.

Index

Appendix A

Office/Exchange behavior, 37

Introduction, 5

Applicability statement, 8

Glossary, 5

Prerequisites/Preconditions, 8

Protocol Overview, 8

References, 7

Relationship to other protocols, 8

Standards assignments, 9

Vendor-extensible fields, 9

Versioning and capability statement, 8

Messages, 9

Message syntax, 9

Transport, 9

Protocol details, 13

Client details, 13

Protocol examples, 17

Finding free/busy messages using e-mail addresses, 19

Free/Busy message, 17

PidTagScheduleInfoFreeBusyBusy calculation, 34

PidTagScheduleInfoFreeBusyMerged calculation, 37

PidTagScheduleInfoMonthsBusy calculation, 32

References

Informative references, 8

Normative references, 7

Security, 37

Index of security parameters, 37

Security considerations for implementers, 37