[MS-OXONOTE]:

Note Object Protocol

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 Open Specification Promise or the Community Promise. 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. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional

development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1		Initial Availability.
6/27/2008	1.0		Initial Release.
8/6/2008	1.01		Revised and edited technical content.
9/3/2008	1.02		Updated references.
12/3/2008	1.03		Updated IP notice.
4/10/2009	2.0		Updated applicable product releases.
7/15/2009	3.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	4.0.0	None	Version 4.0.0 release
5/5/2010	4.0.1	Editorial	Revised and edited the technical content.
8/4/2010	4.1	Minor	Clarified the meaning of the technical content.
11/3/2010	4.2	Minor	Clarified the meaning of the technical content.
3/18/2011	4.2	No change	No changes to the meaning, language, and formatting of the technical content.
8/5/2011	4.3	Minor	Clarified the meaning of the technical content.
10/7/2011	4.3	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	5.0	Major	Significantly changed the technical content.
4/27/2012	5.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	5.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	5.1	Minor	Clarified the meaning of the technical content.
2/11/2013	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	5.1	No Change	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	5.2	Minor	Clarified the meaning of the technical content.
3/16/2015	6.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Note Object Protocol enables the representation of a brief note that functions as the electronic equivalent of a paper sticky note. This protocol extends the Message and Attachment Object Protocol, which is described in [MS-OXCMSG].

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

Attachment object: A set of properties that represents a file, **Message object**, or structured storage that is attached to a Message object and is visible through the attachments table for a Message object.

Folder object: A messaging construct that is typically used to organize data into a hierarchy of objects containing Message objects and folder associated information (FAI) Message objects.

handle: Any token that can be used to identify and access an object such as a device, file, or a window.

long ID (LID): A 32-bit quantity that, in combination with a GUID, defines a **named property**.

Mail User Agent (MUA): A client application that is used to compose and read email messages.

mailbox: A message store that contains email, calendar items, and other **Message objects** for a single recipient.

Message object: A set of properties that represents an email message, appointment, contact, or other type of personal-information-management object. In addition to its own properties, a Message object contains recipient properties that represent the addressees to which it is addressed, and an attachments table that represents any files and other Message objects that are attached to it.

named property: A property that is identified by both a GUID and either a string name or a 32-bit identifier.

named property set: A GUID that groups related named properties into a set.

Note object: A **Message object** that represents a simple text note in a messaging store and that adheres to the property descriptions that are described in [MS-OXONOTE]. A Note object functions as an electronic equivalent of a paper sticky note.

plain text: Text that does not have markup. See also **plain text message body**.

plain text message body: A message body (2) for which the Content-Type value of the Email Text Body header field is "text/plain". A plain text message body can be identified explicitly in the content, or implicitly if it is in a message that is as described in [RFC822] or a message that does not contain a Content-Type header field.

property ID: A 16-bit numeric identifier of a specific attribute (1). A property ID does not include any **property type** information.

property name: A string that, in combination with a property set, identifies a **named property**.

property type: A 16-bit quantity that specifies the data type of a property value.

recipient: An entity that can receive email messages.

remote operation (ROP): An operation that is invoked against a server. Each ROP represents an action, such as delete, send, or query. A ROP is contained in a ROP buffer for transmission over the wire.

Rich Text Format (RTF): Text with formatting as described in [MSFT-RTF].

ROP request: See ROP request buffer.

ROP response: See ROP response buffer.

special folder: One of a default set of **Folder objects** that can be used by an implementation to store and retrieve user data objects.

tagged property: A property that is defined by a 16-bit property ID and a 16-bit property type. The property ID for a tagged property is in the range 0x001 – 0x7FFF. Property IDs in the range 0x8000 – 0x8FFF are reserved for assignment to **named properties**.

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

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

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

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

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

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

[MS-OXOMSG] Microsoft Corporation, "Email Object Protocol".

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

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

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

1.2.2 Informative References

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

[MS-OXOCFG] Microsoft Corporation, "Configuration Information Protocol".

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

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

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

[MSFT-RTF] Microsoft Corporation, "Rich Text Format (RTF) Specification", version 1.9.1, March 2008, http://www.microsoft.com/en-us/download/details.aspx?id=10725

[RFC822] Crocker, D.H., "Standard for ARPA Internet Text Messages", STD 11, RFC 822, August 1982, http://www.ietf.org/rfc/rfc0822.txt

1.3 Overview

The Note Object Protocol allows a user to store in his **mailbox** a simple text note (that is, text with minimal formatting) that functions as the electronic equivalent of a paper sticky note. To represent the sticky note, this protocol defines a **Note object**. The properties that are specific to a Note object contain information about the background color, window location, and size of the note. A Note object is stored in a **Folder object**. The Note Object Protocol also specifies how a Note object is created and manipulated.

The Note Object Protocol extends the Message and Attachment Object Protocol, described in [MS-OXCMSG], in that it defines new properties for a **Message object** and adds constraints to the existing properties of a Message object.

1.4 Relationship to Other Protocols

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

For conceptual background information and overviews of the relationships and interactions between this and other protocols, see [MS-OXPROTO].

1.5 Prerequisites/Preconditions

The Note Object Protocol has the same prerequisites and preconditions as the Message and Attachment Object Protocol, as specified in [MS-OXCMSG].

1.6 Applicability Statement

A client uses this protocol to create and maintain a user's sticky notes.

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

The Note Object Protocol uses the same underlying transport as that used by the Message and Attachment Object Protocol, which is specified in [MS-OXCMSG].

2.2 Message Syntax

A Note object can be created and modified by clients and servers. Except where noted, this section defines constraints under which both clients and servers operate.

A client operates on a Note object by using the Message and Attachment Object Protocol, as specified in [MS-OXCMSG]. How a server operates on a Note object is implementation-dependent, but the results of any such operation MUST be exposed to clients in a manner that is consistent with the Note Object Protocol.

Unless otherwise specified, a Note object adheres to all property constraints specified in [MS-OXPROPS] and all property constraints specified in [MS-OXCMSG].

2.2.1 Note Object Properties

The properties specific to a Note object are defined in section 2.2.1.1 through section 2.2.1.5.

2.2.1.1 PidLidNoteColor Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidLidNoteColor** property ([MS-OXPROPS] section 2.177) specifies the suggested background color of the note. This property MUST be set to one of the values specified in the following table.<1>

Value	Color
0x00000000	Blue
0x00000001	Green
0x00000002	Pink
0x00000003	Yellow
0x00000004	White

2.2.1.2 PidLidNoteWidth Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidLidNoteWidth** property ([MS-OXPROPS] section 2.179) specifies the width of the note's visible window in pixels. The value of this property MUST be greater than zero.

2.2.1.3 PidLidNoteHeight Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidLidNoteHeight** property ([MS-OXPROPS] section 2.178) specifies the height of the note's visible window in pixels. The value of this property MUST be greater than zero.

2.2.1.4 PidLidNoteX Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidLidNoteX** property ([MS-OXPROPS] section 2.180) specifies the distance, in pixels, from the left edge of the screen that a user interface displays the note.

2.2.1.5 PidLidNoteY Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidLidNoteY** property ([MS-OXPROPS] section 2.181) specifies the distance, in pixels, from the top edge of the screen that a user interface displays the note.

2.2.2 Additional Property Constraints

This protocol specifies additional constraints on some Message object properties beyond what is specified in [MS-OXCMSG]. These constraints are specified in sections 2.2.2.1 through 2.2.2.6.

2.2.2.1 Best Body Properties

Best body properties specify the content of the note. The content is a **plain text message body** stored in the **PidTagBody** property ([MS-OXCMSG] section 2.2.1.22.1).<2>

2.2.2.2 PidTagIconIndex Property

Type: **PtypInteger32** ([MS-OXCDATA] section 2.11.1)

The **PidTagIconIndex** property ([MS-OXOMSG] section 2.2.1.10) specifies which icon a user interface is to use when displaying a group of Note objects. The value of this property MUST be 0x00000300 added to the value of the **PidLidNoteColor** property (section 2.2.1.1).

2.2.2.3 PidTagMessageClass Property

Type: **PtypString** ([MS-OXCDATA] section 2.11.1)

The **PidTagMessageClass** property ([MS-OXCMSG] section 2.2.1.3) specifies the type of the Message object. The value MUST be "IPM.StickyNote" or begin with "IPM.StickyNote.", in addition to meeting the criteria specified in [MS-OXCMSG] section 2.2.1.3.

2.2.2.4 PidTagNormalizedSubject Property

Type: **PtypString** ([MS-OXCDATA] section 2.11.1)

The **PidTagNormalizedSubject** property ([MS-OXCMSG] section 2.2.1.10) specifies an abbreviated version of the contents of the note.<3>

2.2.2.5 Recipients

A Note object MUST NOT have **recipients**.

2.2.2.6 Attachment Objects

A Note object MUST NOT have **Attachment objects**.

3 Protocol Details

3.1 Client Details

The client creates and manipulates a Note object and in all other ways operates within the client role as specified in [MS-OXCMSG].

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.

This protocol uses the abstract data model that is specified in [MS-OXCMSG] section 3.1.1 with the following adaptations:

- The Note object is an extension of the Message object.
- A Note object is created in the Notes folder, which is a **special folder**, unless the **Mail User Agent (MUA)** explicitly specifies another folder. For details about special folders, see [MS-OXOSFLD].

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Creating a Note Object

When a user creates a new note, the client creates a Message object as specified in [MS-OXCMSG] section 3.1.4.2, sets properties in accordance with the requirements in section 2.2, and saves the resulting Message object as specified in [MS-OXCMSG] section 3.1.4.3. For details about setting properties, see [MS-OXCPRPT].

3.1.4.2 Modifying a Note Object

When a user opens and modifies an existing note, the client opens the Note object in the same way that it opens any Message object, as specified in [MS-OXCMSG] section 3.1.4.1. The client then modifies any of the properties in accordance with the requirements in section 2.2 and saves the Note object as specified in [MS-OXCMSG] section 3.1.4.3. For details about setting properties, see [MS-OXCPRPT].

3.1.4.3 Deleting a Note Object

When a user deletes a note, the client deletes the Note object in the same way that it deletes any Message object, as specified in [MS-OXCFOLD] section 3.1.4.8.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server processes a client's requests regarding a Note object and in all other ways operates within the server role as specified in [MS-OXCMSG].

3.2.1 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.

This protocol uses the abstract data model that is specified in [MS-OXCMSG] section 3.2.1 with the following adaptations:

- The Note object is an extension of the Message object.
- A Note object is created in the Notes folder, which is a special folder, unless the MUA explicitly specifies another folder. For details about special folders, see [MS-OXOSFLD].

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The server responds to client requests as specified in [MS-OXCMSG] section 3.2.5.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Sample Note Object

Joe creates a Note object, types in his grocery list, and saves it. The following is a description of what a client might do to accomplish Joe's intentions and the responses a server might return.

This example uses both **named properties** and **tagged properties** of a Note object. The **property ID** of a named property is provided by the server. Therefore, before setting or reading any properties of a Note object, the client asks the server to perform a mapping from **property names** or **long IDs (LIDs)** to property IDs. To request this mapping, the client sends a **RopGetPropertyIdsFromNames ROP request** ([MS-OXCROPS] section 2.2.8.1).

The following table lists each named property with its **named property set** GUID and its long ID (LID) or property name. The server's **RopGetPropertyIdsFromNames ROP response** provides the corresponding property IDs, as shown in the subsequent table.

Property	Property set GUID	LID
PidLidNoteColor (section 2.2.1.1)	{0006200E-0000-0000-C000-000000000046}	0x00008B00
PidLidNoteWidth (section 2.2.1.2)	{0006200E-0000-0000-C000-000000000046}	0x00008B02
PidLidNoteHeight (section 2.2.1.3)	{0006200E-0000-0000-C000-000000000046}	0x00008B03
PidLidNoteX (section 2.2.1.4)	{0006200E-0000-0000-C000-000000000046}	0x00008B04
PidLidNoteY (section 2.2.1.5)	{0006200E-0000-0000-C000-000000000046}	0x00008B05

The following table lists the property IDs that might be provided in the server's **RopGetPropertyIdsFromNames** ROP response. The actual property IDs returned are at the discretion of the server.

Property	Property ID
PidLidNoteColor	0x8046
PidLidNoteWidth	0x8047
PidLidNoteHeight	0x8048
PidLidNoteX	0x8049
PidLidNoteY	0x804A

To create a Note object, the client uses the **RopCreateMessage ROP** ([MS-OXCROPS] section 2.2.6.2). The server returns a success code and a **handle** to a Message object.

After Joe has input his content for the Note object, the client transmits the properties of the Note object to the server by using the **RopSetProperties** ROP ([MS-OXCROPS] section 2.2.8.6). The properties that are set on the Note object are shown in the following table. For information about **property types** in the following table, see[MS-OXCDATA] section 2.11.1.

Property	Property ID	Property type	Value
PidLidNoteColor	0x8046	0x0003	0x00000003

Property	Property ID	Property type	Value
		(**PtypInteger32**)	
PidLidNoteWidth	0x8047	0x0003	0x000000C8
PidLidNoteHeight	0x8048	0x0003	0x000000A6
PidLidNoteX	0x8049	0x0003	0x0000006E
PidLidNoteY	0x804A	0x0003	0x0000006E
PidTagIconIndex ([MS-OXOMSG] section 2.2.1.10)	0x1080	0x0003	0x00000303
PidTagMessageClass ([MS-OXCMSG] section 2.2.1.3)	0x001A	0x001F (**PtypString**)	"IPM.StickyNote"
PidTagNormalizedSubject ([MS-OXCMSG] section 2.2.1.10)	0x0E1D	0x001F	"Grocery List"
PidTagSubjectPrefix ([MS-OXCMSG] section 2.2.1.9)	0x003D	0x001F	"" (null)
PidTagBody ([MS-OXCMSG] section 2.2.1.48.1)	0x1000	0x001F	"Grocery List: Celery Broccoli"

When Joe is ready to save his changes, the client commits the properties on the server by using the **RopSaveChangesMessage** ROP ([MS-OXCROPS] section 2.2.6.3) and then releases the Note object by using the **RopRelease** ROP ([MS-OXCROPS] section 2.2.15.3).

The values of some properties of the Message object will change during the execution of the **RopSaveChangesMessage** ROP, but the properties specified in this document will not change.

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations specific to the Note Object Protocol. General security considerations pertaining to the underlying transport apply, as described in [MS-OXCMSG].

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

- Microsoft Exchange Server 2003
- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016 Preview

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

<1> Section 2.2.1.1: Microsoft Office Outlook 2003 Service Pack 3 (SP3) will always use the **PidLidNoteColor** property to determine the background color, regardless of the existence or value of the **PidNameKeywords** property ([MS-OXCMSG] section 2.2.1.17). Microsoft Office Outlook 2007 Service Pack 1 ignores the **PidLidNoteColor** property if the item has the **PidNameKeywords** property set also. In that case, the background color is the color associated with the first keyword listed, as described in [MS-OXOCFG].

<2> Section 2.2.2.1: Office Outlook 2003 SP3 and Office Outlook 2007 SP1 set encapsulated **plain text** as a **Rich Text Format (RTF)** for the message body. For more information, see [MS-OXRTFEX] and [MS-OXCMSG].

<3> Section 2.2.2.4: Office Outlook 2007 always sets this property to the first line of the message body.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

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

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

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

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
6 Appendix A: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

8 Index

R

S

T

V