

[MS-ASCON]: ActiveSync Conversations 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.aspx>). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) 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 (beta) 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
04/10/2009	0.1.0	Major	Initial Availability.
07/15/2009	1.0.0	Major	Revised and edited for technical content.
11/04/2009	2.0.0	Major	Updated and revised the technical content.
02/10/2010	2.1.0	Minor	Updated the technical content.
05/05/2010	3.0.0	Major	Updated and revised the technical content.

Table of Contents

1 Introduction	5
1.1 Glossary	5
1.2 References	5
1.2.1 Normative References	5
1.2.2 Informative References	6
1.3 Overview	6
1.4 Relationship to Other Protocols	6
1.5 Prerequisites/Preconditions	6
1.6 Applicability Statement	6
1.7 Versioning and Capability Negotiation	6
1.8 Vendor-Extensible Fields	6
1.9 Standards Assignments	6
2 Messages	7
2.1 Transport	7
2.2 Message Syntax	7
2.2.1 Complex Types	7
2.2.1.1 ItemOperations Command Complex Types	8
2.2.1.1.1 Move	8
2.2.1.1.2 Move.Options	8
2.2.1.1.3 Response.Move	8
2.2.1.2 AirSyncBase Namespace Complex Types	8
2.2.1.2.1 BodyPart	8
2.2.1.2.2 BodyPartPreference	9
2.2.2 Elements	9
2.2.2.1 E-mail Class Elements	10
2.2.2.1.1 ConversationId	10
2.2.2.1.2 ConversationIndex	10
2.2.2.2 GetItemEstimate Command Elements	11
2.2.2.2.1 Collections.Collection.ConversationMode	11
2.2.2.3 ItemOperations Command Elements	11
2.2.2.3.1 Move.ConversationId	11
2.2.2.3.2 Move.DstFldId	11
2.2.2.3.3 Move.Options.MoveAlways	12
2.2.2.3.4 Response.Move.Status	12
2.2.2.3.5 Response.Move.ConversationId	12
2.2.2.4 Search Command Elements	12
2.2.2.4.1 Store.Query.ConversationId	12
2.2.2.5 Sync Command Elements	13
2.2.2.5.1 Collections.Collection.ConversationMode	13
3 Protocol Details	14
3.1 Client Details	14
3.1.1 Abstract Data Model	14
3.1.2 Timers	14
3.1.3 Initialization	14
3.1.4 Higher-Layer Triggered Events	14
3.1.4.1 Deleting a Conversation	14
3.1.4.2 Flagging a Conversation for Follow-up	14
3.1.4.3 Marking a Conversation as Read or Unread	15

3.1.4.4	Ignoring a Conversation	15
3.1.4.5	Moving a Conversation from the Current Folder	15
3.1.4.6	Setting up a Conversation to be Moved Always	15
3.1.4.7	Finding a Conversation	15
3.1.4.8	Synchronizing a Conversation	16
3.1.4.9	Applying a Conversation-based Filter	16
3.1.4.10	Requesting a Message Part	16
3.1.5	Message Processing Events and Sequencing Rules	16
3.1.6	Timer Events	16
3.1.7	Other Local Events	16
3.2	Server Details	17
3.2.1	Abstract Data Model	17
3.2.2	Timers	17
3.2.3	Initialization	17
3.2.4	Higher-Layer Triggered Events	17
3.2.5	Message Processing Events and Sequencing Rules	17
3.2.5.1	Processing a Sync Command	17
3.2.5.1.1	Deleting a Conversation	17
3.2.5.1.2	Marking a Conversation as Read or Unread	17
3.2.5.1.3	Flagging a Conversation for Follow-up	18
3.2.5.2	Processing a GetItemEstimate Command	18
3.2.5.3	Processing an ItemOperations Command	18
3.2.5.3.1	Ignoring a Conversation	18
3.2.5.3.2	Always Moving a Conversation	18
3.2.5.4	Processing a MoveItems Command	18
3.2.5.5	Processing a Search Command	18
3.2.5.6	Filtering	19
3.2.5.7	Sending a Message Part	19
3.2.6	Timer Events	20
3.2.7	Other Local Events	20
4	Protocol Examples	21
4.1	Synchronization From the Server	21
4.2	Ignoring a Conversation	22
5	Security	23
5.1	Security Considerations for Implementers	23
5.2	Index of Security Parameters	23
6	Appendix A: Product Behavior	24
7	Change Tracking	25
8	Index	29

1 Introduction

This document specifies the ActiveSync Conversations protocol, which is an **XML**-based format that is used to improve the ways in which e-mail messages are triaged when they are displayed in **conversation** view.

1.1 Glossary

The following terms are defined in [\[MS-OXGLOS\]](#):

class
conversation
conversation ID
conversation index
GUID
Hypertext Markup Language (HTML)
message
message part
WAP Binary XML (WBXML)
XML
XML schema

The following terms are specific to this document:

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

1.2 References

1.2.1 Normative References

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

[MS-ASCMD] Microsoft Corporation, "[ActiveSync Command Reference Protocol Specification](#)", December 2008.

[MS-ASDTYPE] Microsoft Corporation, "[ActiveSync Data Types](#)", December 2008.

[MS-ASEMAIL] Microsoft Corporation, "[ActiveSync E-Mail Class Protocol Specification](#)", December 2008.

[MS-ASWBXML] Microsoft Corporation, "[ActiveSync WAP Binary XML \(WBXML\) Protocol Specification](#)", December 2008.

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

[MS-OXGLOS] Microsoft Corporation, "[Exchange Server Protocols Master Glossary](#)", April 2008.

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

1.2.2 Informative References

None.

1.3 Overview

The ActiveSync Conversations protocol is an XML-based format that is used to improve the ways in which e-mail messages are triaged, allowing a user to view a series of send-response e-mail messages as a single representation, called a conversation.

A conversation appears in a message folder as one unit and allows the user to read the series of related e-mail messages in a single effort. Each e-mail message is assigned a **conversation ID** that is used to identify the conversation to which the e-mail message belongs.

1.4 Relationship to Other Protocols

The ActiveSync Conversations protocol consists of a series of XML elements that are embedded inside a command request or a command response. For details about command requests and responses, see [\[MS-ASCMD\]](#). The **WAP Binary XML (WBXML)**, specified in [\[MS-ASWBXML\]](#), is used to transmit the XML markup that constitutes the request body or the response body.

The ActiveSync Conversations protocol defines elements and complex types according to the data type definitions that are specified in [\[MS-ASDTYPE\]](#).

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

This protocol is applicable in scenarios in which a client needs to synchronize its e-mail messages and files with a server and wants to present a view in which e-mail messages are grouped by conversation rather than listed serially.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The ActiveSync Conversations protocol consists of a series of XML elements that are embedded inside a command request or a command response. The XML markup that constitutes the request body or the response body is transmitted between client and server by using WAP Binary XML (WBXML), as specified in [\[MS-ASWBXML\]](#).

2.2 Message Syntax

The XML markup that is used by the ActiveSync Conversations protocol MUST be well-formed XML, as specified in [\[XML\]](#).

The XML elements that are used by the ActiveSync Conversations protocol are embedded inside a request or response for the following commands:

GetItemEstimate

ItemOperations

MoveItems

Search

Sync

For details about the requests and responses for these commands, see [\[MS-ASCMD\]](#).

The complex types and elements of the ActiveSync Conversations protocol are defined in the following namespaces: AirSync, AirSyncBase, Email2, ItemOperations, and Search.

2.2.1 Complex Types

The following table summarizes the set of common **XML schema** complex types that are defined by this specification for the **ItemOperations**, **Search**, and **Sync** commands. For more details about these commands, see [\[MS-ASCMD\]](#) sections [2.2.1.8](#), [2.2.1.14](#), and [2.2.1.19](#), respectively.

Complex Type	Description
Move	Used in an ItemOperations command request to indicate that a conversation is to be moved.
Move.Options	Used in an ItemOperations command request to specify the options for moving a conversation.
Response.Move	Used in an ItemOperations command response to specify the results of the attempt to move a conversation.
BodyPart	Used in ItemOperations , Search , and Sync command responses to encapsulate a message part and its meta-data.
BodyPartPreference	Used in ItemOperations , Search , and Sync command requests to specify preferences for receiving a message part from the server.

2.2.1.1 ItemOperations Command Complex Types

The following complex types are defined in the **ItemOperations** namespace.

2.2.1.1.1 Move

The **Move** type indicates that a conversation is to be moved from all folders to a destination folder. The **Move** type is a required **container** type in an **ItemOperations** command request. It contains the following child elements and type:

- **ConversationId** element — see section [2.2.2.3.1](#)
- **DstFldId** element — see section [2.2.2.3.2](#)
- **Options** type — see section [2.2.1.1.2](#)

The **container** type is specified in [\[MS-ASDTYPE\]](#) section 2.8.

2.2.1.1.2 Move.Options

The **Move.Options** type specifies the options for the **Move** type. The **Move.Options** type is an optional **container** type in an **ItemOperations** command request. It contains the following child element:

- **MoveAlways** element — see section [2.2.2.3.3](#)

The **container** type is specified in [\[MS-ASDTYPE\]](#) section 2.8.

2.2.1.1.3 Response.Move

The **Response.Move** type specifies the results of the attempt to move a conversation. The **Response.Move** type is a required **container** type in an **ItemOperations** command response. It contains the following child elements:

- **ConversationId** element — see section [2.2.2.3.5](#)
- **Status** element — see section [2.2.2.3.4](#)

The **container** type is specified in [\[MS-ASDTYPE\]](#) section 2.8.

2.2.1.2 AirSyncBase Namespace Complex Types

The following complex types are defined in the AirSyncBase namespace. For more details about the AirSyncBase namespace, see [\[MS-ASAIRS\]](#).

2.2.1.2.1 BodyPart

The **BodyPart** type encapsulates a message part and its meta-data in a **Sync**, **ItemOperations**, or **Search** command response. The **BodyPart** type is a child of the **ApplicationData** type in a **Sync** command response, and a child of the **Properties** type in an **ItemOperations** command response and a **Search** command response.

The **BodyPart** type is a **container** type. It has the following child elements:

<Status>

<Type>
<EstimatedDataSize>
<Truncated>
<Data>
<Preview>

The **BodyPart** type and its child elements are further specified in [\[MS-ASAIRS\]](#). The **container** type is specified in [\[MS-ASDTYPE\]](#) section 2.8.

2.2.1.2.2 BodyPartPreference

The **BodyPartPreference** type specifies the client's preferences for receiving a message part from the server. The **BodyPartPreference** type is a child of the **Options** type in an **ItemOperations**, **Search**, or **Sync** command request.

The **BodyPartPreference** type is a **container** type. It has the following child elements:

- <Type>
- <TruncationSize>
- <AllOrNone>
- <Preview>

The **BodyPartPreference** type and its child elements are further specified in [\[MS-ASAIRS\]](#). The **container** type is specified in [\[MS-ASDTYPE\]](#) section 2.8.

2.2.2 Elements

The following tables summarize the set of common XML schema elements that are defined by this specification for the E-mail **class**, **GetItemEstimate** command, **ItemOperations** command, **Search** command, and **Sync** command. For details about the E-mail class, see [\[MS-ASEMAIL\]](#). For details about the commands, see [\[MS-ASCMD\]](#) sections [2.2.1.7](#), [2.2.1.8](#), [2.2.1.14](#), and [2.2.1.19](#).

Elements MUST NOT have child elements in either the command request or command response.

Element	Description
ConversationId	Used by the E-mail class to specify the conversation ID for an e-mail message.
ConversationIndex	Used by the E-mail class to specify the conversation index for an e-mail message.
Collections.Collection.ConversationMode (GetItemEstimate command)	Used by the GetItemEstimate command to enable conversation-based filtering of item estimates.
Move.ConversationId	Used by the ItemOperations command to specify the conversation ID of the conversation that is to be moved.
Move.DstFldId	Used by the ItemOperations command to specify the destination folder, which is the folder to which the

Element	Description
	conversation is moved.
Move.Options.MoveAlways	Used by the ItemOperations command to set up the conversation to be always moved.
Response.Move.Status	Used by the ItemOperations command to specify the status of the move action.
Response.Move.ConversationId	Used by the ItemOperations command to specify the conversation ID of the conversation that is moved.
Store.Query.ConversationId	Used by the Search command to specify the conversation ID of the conversation for which to search.
Collections.Collection.ConversationMode (Sync command)	Used by the Sync command to enable conversation-based filtering and synchronization of conversation-based properties.

2.2.2.1 E-mail Class Elements

The following elements are defined in the **Email2** namespace.

2.2.2.1.1 ConversationId

The <ConversationId> element specifies a unique identifier for a conversation. This element is a required child element of the **ApplicationData** type in the **Sync** command response, as specified in [\[MS-ASEMAIL\]](#) section 2.2.

The value of this element is a **byte array**, as specified in [\[MS-ASDTYPE\]](#) section 2.11. The <ConversationId> element is not present if there is no conversation ID associated with the message.

The client MUST NOT change the **ConversationId** value.

2.2.2.1.2 ConversationIndex

The <ConversationIndex> element specifies the conversation index for an e-mail message. This element is a required child element of the **ApplicationData** type in the **Sync** command response, as specified in [\[MS-ASEMAIL\]](#) section 2.2.

The value of this element is a **byte array**, as specified in [\[MS-ASDTYPE\]](#) section 2.11. The value comprises a set of timestamps, which can be used by a client to generate a tree-view of a conversation. The first timestamp identifies the date and time when the message was originally sent by the server. Each additional timestamp specifies the difference between the current time and the time specified by the first timestamp. Additional timestamps are added when the message is forwarded or replied to.

The first timestamp is a **FILETIME** structure with the high-order 5 bytes containing the timestamp and the low-order 3 bytes set to zeros. The **FILETIME** structure is specified in [\[MS-DTYP\]](#) section 2.3.1.

Each additional timestamp is composed of 5 bytes, as follows:

- Delta-code: 1 bit representing the difference between the current time and the time stored in the first timestamp. If the difference is less than 625 days, this bit is 0; otherwise, this bit is 1.

- Time-delta: 31 bits specifying the difference between the current time and the time in the first timestamp, expressed in **FILETIME** units. The time-delta is derived from the 64-bit value that specifies the first timestamp. The delta-code determines how the time-delta is derived. If the delta-code is zero, then the server discards the high-order 15 bits and the low-order 18 bits of the 64-bit value. If the delta-code is 1, then the server discards the high-order 10 bits and the low-order 23 bits.
- Unused: 8 bits specifying a random number that is generated by using an implementation-specific algorithm.

The content of the <ConversationIndex> element is transferred as a **byte array** within the WBXML tags.

The client MUST NOT change the conversation index.

2.2.2.2 GetItemEstimate Command Elements

The following elements are defined in the **AirSync** namespace.

2.2.2.2.1 Collections.Collection.ConversationMode

The **ConversationMode** element enables or disables conversation-based filtering of item estimates. This element is an optional child element of the **Collections.Collection** type in the **GetItemEstimate** command request. For details about the **Collections.Collection** type, see [\[MS-ASCMD\]](#) section 2.2.1.7.1.3.

The value of this element is a **Boolean**, as specified in [\[MS-ASDTYPE\]](#) section 2.3. The value 1 enables conversation-based filtering of item estimates; the value zero disables it. If this element is present without a value, the default is 1.

2.2.2.3 ItemOperations Command Elements

The following elements are defined in the **ItemOperations** namespace.

2.2.2.3.1 Move.ConversationId

The **ConversationId** element specifies the conversation ID of the conversation that is to be moved. This element is a required child element of the **Move** type in the **ItemOperations** command request.

The value of this element is a **byte array**, as specified in [\[MS-ASDTYPE\]](#) section 2.11.

2.2.2.3.2 Move.DstFldId

The <DstFldId> element specifies the destination folder, which is the folder to which the conversation is moved. This element is a required child element of the **Move** type in the **ItemOperations** command request.

The value of this element is a **string**, as specified in [\[MS-ASDTYPE\]](#) section 2.1. The destination folder MUST be of type "IPF.Note". For more information about folder types, see [\[MS-OXOSFLD\]](#) section 2.2.5.

2.2.2.3.3 Move.Options.MoveAlways

The **MoveAlways** element indicates whether a conversation is to be always moved. When a conversation is set to be moved always, all e-mail messages in the conversation, including all future e-mail messages for that conversation, are moved from all folders to a destination folder. This element is an optional child element of the **Move.Options** type in the **ItemOperations** command request.

This element is a flag, which does not have a value. If this element is present, the conversation is set to be moved always.

2.2.2.3.4 Response.Move.Status

The <Status> element specifies the status of the move action. This element is a required child element of the **Response.Move** type in the **ItemOperations** command response.

The value of this element is an **integer**, as specified in [\[MS-ASDTYPE\]](#) section 2.2. The following table lists some of the status values and their meanings.

Value	Meaning
1	Success. The server successfully completed the operation.
2	Protocol error. The XML is not valid.
3	Server error. There was a complete or partial failure of the operation.
6	Not Found. The conversation or destination folder does not exist.
105	Invalid Combination of IDs. The destination folder cannot be the Recipient Information Cache.
155	Protocol error. The Move.Options type does not contain a <MoveAlways> element.
156	Action not supported. The destination folder MUST be of type "IPF.Note". For more information about folder types, see [MS-OXOSFLD] section 2.2.5.

2.2.2.3.5 Response.Move.ConversationId

The **ConversationId** element specifies the conversation ID of the conversation that is moved. This element is a required child element of the **Response.Move** type in the **ItemOperations** command response.

The value of this element is a **byte array**, as specified in [\[MS-ASDTYPE\]](#) section 2.11.

2.2.2.4 Search Command Elements

The following elements are defined in the **Search** namespace.

2.2.2.4.1 Store.Query.ConversationId

The **ConversationId** element specifies the conversation ID of the conversation for which to search. This element is an optional child element of the **Store.Query** type in the **Search** command request. For details about the **Store.Query** type, see [\[MS-ASCMD\]](#) section 2.2.1.14.1.2.

The value of this element is a **byte array**, as specified in [\[MS-ASDTYPE\]](#) section 2.11.

2.2.2.5 Sync Command Elements

The following elements are defined in the **AirSync** namespace.

2.2.2.5.1 Collections.Collection.ConversationMode

The **ConversationMode** element enables or disables conversation-based filtering and synchronization of conversation-based properties. This element is an optional child element of the **Collections.Collection** type in the **Sync** command request. For details about the **Collections.Collection** type, see [\[MS-ASCMD\]](#) section 2.2.1.19.1.10.

The value of this element is a **Boolean**, as specified in [\[MS-ASDTYPE\]](#) section 2.3. The value 1 enables conversation-based filtering and synchronization of conversation-based properties; the value zero disables it. If this element is present without a value, the default is 1.

PRELIMINARY

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

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

Command request: A WBXML-formatted message that adheres to the command schemas specified in [\[MS-ASCMD\]](#).

E-mail messages: Every e-mail message includes the following to support conversations:

- Conversation ID: A unique value that is associated with a conversation. This value is specified by the **ConversationId** element that is included in the E-mail class. For details about the E-mail class, see [\[MS-AEMAIL\]](#).
- Conversation index: An index that is used by clients to generate a conversation tree view. This value is specified by the **ConversationIndex** element that is included in the E-mail class. For details about the E-mail class, see [\[MS-AEMAIL\]](#).

The server creates a conversation ID and a conversation index on the e-mail item when the user sends an e-mail message. The client does not change the conversation ID or the conversation index.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Deleting a Conversation

When a conversation is deleted, all e-mail messages that are in the conversation are moved from the current folder to the Deleted Items folder. Future e-mail messages for the same conversation are not affected.

To delete a conversation, the client sends a **Sync** command request that contains a **Delete** element for each item in the conversation. For more details about the **Delete** element in the **Sync** command request, see [\[MS-ASCMD\]](#) section 2.2.1.19.1.18.

3.1.4.2 Flagging a Conversation for Follow-up

When a conversation is flagged for follow-up, the most recent e-mail message that is in the conversation and that is in the current folder is flagged. Clearing a flag on a conversation will clear flags on all e-mail messages that are in the conversation and that are in the current folder. Marking

a flagged conversation as complete will mark all flagged e-mail messages that are in the conversation and that are in the current folder as complete.

To set a flag on a conversation, clear a flag on a conversation, or mark a flagged conversation as complete, the client sends a **Sync** command request that contains a **Change** element, as specified in [\[MS-ASCMD\]](#) section 2.2.1.19.1.8.

3.1.4.3 Marking a Conversation as Read or Unread

When a conversation is marked as read or unread, all e-mail messages that are in the conversation and that are in the current folder are marked as such.

To mark a conversation as read or unread, the client sends a **Sync** command request that contains a **Change** element, as specified in [\[MS-ASCMD\]](#) section 2.2.1.19.1.8.

3.1.4.4 Ignoring a Conversation

When a conversation is ignored, all e-mail messages in the conversation, including all future e-mail messages for that conversation, are moved from all folders to the Deleted Items folder.

To ignore a conversation, the client sends an **ItemOperations** command request that contains a **Move** type and its child elements, as specified in section [2.2.1.1.1](#) of this document. The **Move.Options.MoveAlways** element MUST be present and the **Move.DstFldId** element MUST contain the ID of the Deleted Items folder. Multiple **Move** types, one for each conversation to be moved, can be included within one **ItemOperations** request. In this case, the **Move** types are processed in the order specified. For details about the **ItemOperations** command request, see [\[MS-ASCMD\]](#) section 2.2.1.8.2.

3.1.4.5 Moving a Conversation from the Current Folder

When a conversation is moved from the current folder to another folder, all e-mail messages that are in the conversation are moved from the current folder to the destination folder.

To move a conversation from the current folder to a destination folder, the client sends a **MoveItems** command request, as specified in [\[MS-ASCMD\]](#) section 2.2.1.10.1.

3.1.4.6 Setting up a Conversation to be Moved Always

When a conversation is set to be moved always, all e-mail messages in the conversation, including all future e-mail messages for that conversation, are moved from all folders to a destination folder.

To set a conversation to be moved always, the client sends an **ItemOperations** command request that contains a **Move** type and its child elements, as specified in section [2.2.1.1.1](#) of this document. The **Move.Options.MoveAlways** element MUST be present. The client MUST NOT specify the Outbox folder, the Drafts folder, or the Recipient Information Cache as the destination folder. For details about the **ItemOperations** command request, see [\[MS-ASCMD\]](#) section 2.2.1.8.2.

3.1.4.7 Finding a Conversation

Searching for a particular conversation will search across all folders for all e-mail messages that are in the conversation.

To search for a conversation, the client sends a **Search** command request with the **Store.Query.ConversationId** element, which is specified in section [2.2.2.4.1](#) of this document. The **Store.Query.ConversationId** element can be used in conjunction with other child elements of

the **Store.Query** type. The client MUST scope the query to the Email class by setting the **Store.Name** element to "Mailbox".

For details about the **Search** command request, see [\[MS-ASCMD\]](#) section 2.2.1.14.1. For details about the **Store.Name** element, see [\[MS-ASCMD\]](#) section 2.2.1.14.1.1. For details about the **Store.Query** type and its child elements, see [\[MS-ASCMD\]](#) section 2.2.1.14.1.2.

3.1.4.8 Synchronizing a Conversation

When a conversation is synchronized, all e-mail messages that are part of the conversation and that are in the specified folder are synchronized.

To synchronize a conversation, the client sends a **Sync** command request with a **Collections.Collection.ConversationMode** element for the particular collection to be synchronized.

3.1.4.9 Applying a Conversation-based Filter

Conversation-based filtering augments the date-based filtering. For details about date-based filtering, see [\[MS-ASCMD\]](#) section 2.2.1.19.1.23.

When a conversation-based filter is applied to a synchronization of the current folder, the complete conversation is retrieved if any e-mail message in the conversation falls within the date-based filter.

To apply a conversation-based filter to a synchronization, the client includes the **Collections.Collection.ConversationMode** element in a **Sync** command request. For details about the **Sync** command request, see [\[MS-ASCMD\]](#) section 2.2.1.19.1.

A conversation-based filter can also be applied to the **GetItemEstimate** command to get an estimate of the items that both meet the filter criteria and need to be synchronized. The client can apply the filter by including the **Collections.Collection.ConversationMode** element in a **GetItemEstimate** command request. For details about the **GetItemEstimate** command request, see [\[MS-ASCMD\]](#) section 2.2.1.7.1.

3.1.4.10 Requesting a Message Part

When a client synchronizes, searches, or fetches an e-mail **message**, the client can choose to receive a message part by including the **BodyPartPreference** type, which is specified in section [2.2.1.2.2](#), in the **Sync**, **Search**, or **ItemOperations** command request. The value of the <Type> element MUST be 2 to specify **HTML** as the encoding format.

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

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.

Command response: A WBXML-formatted message that adheres to the command schemas specified in [\[MS-ASCMD\]](#).

E-mail messages: Every e-mail item includes the following to support conversations:

- **Conversation ID:** A unique value that is associated with a conversation. This value is specified by the **ConversationId** element that is included in the E-mail class. For details about the E-mail class, see [\[MS-AEMAIL\]](#).
- **Conversation Index:** An index that is used by clients to generate a conversation tree view. This value is specified by the **ConversationIndex** element that is included in the E-mail class. For details about the E-mail class, see [\[MS-AEMAIL\]](#).

The server creates a conversation ID and a conversation index on the e-mail item when the user sends an e-mail Message. The client does not change the conversation ID or the conversation index.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

The server creates a conversation ID and a conversation index on the e-mail item when the user sends an e-mail message.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Processing a Sync Command

3.2.5.1.1 Deleting a Conversation

The server moves all e-mail messages that are in the conversation from the current folder to the Deleted Items folder. The server does not move future e-mail messages for the conversation.

The server sends a **Sync** response, as specified in [\[MS-ASCMD\]](#) section 2.2.1.19.2.

3.2.5.1.2 Marking a Conversation as Read or Unread

The server marks all e-mails that are in the conversation and that are in the current folder as either read or unread, whichever is specified in the client's request (see section [3.1.4.3](#)).

The server sends a **Sync** response, as specified in [\[MS-ASCMD\]](#) section 2.2.1.19.2.

3.2.5.1.3 Flagging a Conversation for Follow-up

If a conversation is flagged for follow-up, the server flags the most recent e-mail message that is in the conversation and that is in the current folder. If a flag is cleared on a conversation, the server clears flags on all e-mail messages that are in the conversation and that are in the current folder. If a flagged conversation is marked as complete, the server marks all flagged e-mail messages that are in the conversation and that are in the current folder as complete.

The server sends a **Sync** response, as specified in [\[MS-ASCMD\]](#) section 2.2.1.19.2.

3.2.5.2 Processing a GetItemEstimate Command

When a conversation-based filter is applied to the **GetItemEstimate** command, the server sends an estimate of the items that meet the filter criteria and need to be synchronized.

In the event of failure, the server sends the following status code. For details about the **GetItemEstimate** response, see [\[MS-ASCMD\]](#) section 2.2.1.7.2.

Value	Meaning
4	Protocol error. The conversation-based filter cannot be applied to a folder that is not of the E-mail class.

3.2.5.3 Processing an ItemOperations Command

3.2.5.3.1 Ignoring a Conversation

When a conversation is ignored, the server moves all e-mail messages in the conversation, including all future e-mail messages for that conversation, from all folders to the Deleted Items folder.

The server's response includes the **Response.Move.Status** element, which contains one of the values specified in section [2.2.2.3.4](#), and the **Response.Move.ConversationId** element.

3.2.5.3.2 Always Moving a Conversation

When a conversation is set to be moved always, the server moves all e-mail messages in the conversation, including all future e-mail messages for that conversation, from all folders to a destination folder.

The server's response includes the **Response.Move.Status** element, which contains one of the values specified in section [2.2.2.3.4](#), and the **Response.Move.ConversationId** element.

3.2.5.4 Processing a MoveItems Command

The server moves all e-mail messages that are in the conversation from the current folder to the destination folder. The server sends a **MoveItems** response, as specified in [\[MS-ASCMD\]](#) section 2.2.1.10.2.

3.2.5.5 Processing a Search Command

The server searches across all folders for all e-mail messages that are in the conversation and returns this set of e-mail messages. For details about the **Search** command response, see [\[MS-ASCMD\]](#) section 2.2.1.14.2.

3.2.5.6 Filtering

If an individual e-mail message is moved or deleted, and, as a result, the rest of the messages in the conversation fall out of filter, the server SHOULD send soft deletes (**Sync** command response) for those messages only during the aging-off process. The aging-off process is explained in the following paragraph. For details about the **Sync** command and soft deletes, see [MS-ASCMD] sections [2.2.1.19](#) and [2.2.1.19.2.11](#), respectively.

The aging-off process is the process in which the server deletes objects from the client that are older than the given time-window. The time-window is specified by the client in the <FilterType> element of the **Sync** command request. For more details about this element, see [MS-ASCMD] section 2.2.1.19.1.23. The server typically performs the aging-off process daily at midnight, but the time and frequency of execution is implementation-dependent. An example of how the aging-off process is applied to conversations is as follows: Suppose that the client specifies a three-day time-window. If any e-mail within a conversation is less than three days old, all e-mails (going back in time to the oldest item in the mailbox) within that conversation will be synchronized to the client. Once the newest e-mail within the conversation becomes older than three days, the server will send soft deletes for all of the e-mails that are within the conversation.

3.2.5.7 Sending a Message Part

If the client's **Sync**, **Search**, or **ItemOperations** command request includes the **BodyPartPreference** type, then the server uses the **BodyPart** type to encapsulate the message part in the response. The **BodyPart** type is not present in the response when the client did not request the message part, as specified in section [3.1.4.10](#).

The client's preferences affect the server's response as follows:

- If the size of the message part exceeds the value specified in the <TruncationSize> element of the request, then the server truncates the message part. The server includes the <Truncated> element and the <EstimatedDataSize> element when it truncates the message part.
- If a value other than 2 is specified in the <Type> element of the request, then the server returns a status value of 164.
- If the client enables conversation-based filtering (section [3.1.4.9](#)), then the server returns a message part for each of the conversation's e-mails that falls outside of the date-based filter.

The **BodyPart** type and the **Body** type can co-exist in the response. The presence or absence of **BodyPart** or **Body** in the response depends on the preferences specified in the request, as summarized in the following table.

Request Contains	Response Contains
Neither BodyPreference nor BodyPartPreference complex types	Only Body complex type
Only BodyPreference complex type	Only Body complex type
Only BodyPartPreference complex type	Only BodyPart complex type
Both BodyPreference and BodyPartPreference complex types	Both Body and BodyPart complex types

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

PRELIMINARY

4 Protocol Examples

4.1 Synchronization From the Server

The following example shows the server returning an e-mail message. Note that the conversation ID and conversation index are included in the **ApplicationData** node. A server can choose any name for a namespace and then map its chosen name to the actual namespace name. This example shows alternate namespace names being used by the server. [<1>](#) These alternate names map to the actual namespaces as follows:

POOMMAIL maps to Email

POOMMAIL2 maps to Email2

Response from the server:

```
<?xml version="1.0" encoding="utf-8"?>
<Sync xmlns:A="POOMMAIL:" xmlns:B="AirSyncBase:"
xmlns:C="POOMMAIL2:" xmlns="AirSync:">
  <Collections>
    <Collection>
      <SyncKey>1601897837</SyncKey>
      <CollectionId>7</CollectionId>
      <Status>1</Status>
      <Commands>
        <Add>
          <ServerId>7:1</ServerId>
          <ApplicationData>
            <A:To>"deviceuser" &lt;someone@example.com&gt;</A:To>
            <A:From>"deviceuser2" &lt;someone2@example.com&gt;</A:From>
            <A:Subject>Test report</A:Subject>
            <A:DateReceived>2009-03-21T07:04:26.948Z</A:DateReceived>
            <A:DisplayTo>deviceuser</A:DisplayTo>
            <A:ThreadTopic>Test report</A:ThreadTopic>
            <A:Importance>1</A:Importance>
            <A:Read>1</A:Read>
            <B:Body>
              <B:Type>1</B:Type>
              <B:EstimatedDataSize>100</B:EstimatedDataSize>
              <B:Truncated>1</B:Truncated>
              <B:Data>Test data</B:Data>
            </B:Body>
            <A:MessageClass>IPM.Note</A:MessageClass>
            <A:InternetCPID>20127</A:InternetCPID>
            <A:Flag/>
            <A:ContentClass>urn:content-classes:message</A:ContentClass>
            <B:NativeBodyType>2</B:NativeBodyType>
            <C:ConversationId>BBA4726D4399D44C83297D4BD904ED2D</C:ConversationId>
            <C:ConversationIndex>01C9A9F345</C:ConversationIndex>
            <A:Categories/>
          </ApplicationData>
        </Add>
      </Commands>
    </Collection>
  </Collections>
```

```
</Sync>
```

4.2 Ignoring a Conversation

The following example shows the client's request to ignore a conversation and the server's response.

Request from the client:

```
<ItemOperations>
  <Move>
    <ConversationId>...</ConversationId>
    <DstFldId>DeletedItems-Folder-ID</DstFldId>
    <Options>
      <MoveAlways/>
    </Options>
  </Move>
</ItemOperations>
```

Response from the server:

```
<ItemOperations>
  <Response>
    <Status>1</Status>
    <Move>
      <Status>1</Status>
      <ConversationId>...</ConversationId>
    </Move>
  </Response>
</ItemOperations>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

PRELIMINARY

6 Appendix A: Product Behavior

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

- Microsoft® Exchange Server 2010
- Microsoft® Exchange Server 2010 SP1 Beta

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

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

[<1> Section 4.1](#): Exchange uses alternate names that are prefixed with "POOM", which stands for "Pocket Outlook Object Model".

7 Change Tracking

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- A protocol is deprecated.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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

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

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

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

- New content added.
- Content update.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.

- Content removed for template compliance.
- Obsolete document removed.

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
1.1 Glossary	54765 Added new term "message part".	N	New content added.
1.1 Glossary	Added terms "Hypertext Markup Language" and "message".	N	Content update.
1.3 Overview	Changed "the Inbox" to "a message folder".	N	Content update.
1.3 Overview	Updated the section title.	N	Content updated for template compliance.
2.2 Message Syntax	54765 Added the AirSyncBase namespace to the list of namespaces.	N	Content update.
2.2.1 Complex Types	54765 Added the BodyPart and BodyPartPreference complex types to the table and updated the introductory paragraph with references to Search and Sync commands. Updated the descriptions of the Move, Move.Options, and Response.Move complex types.	Y	Content update.
2.2.1.2 AirSyncBase Namespace Complex Types	54765 New section.	Y	New content added.
2.2.1.2.1 BodyPart	54765 New section.	Y	New content added.
2.2.1.2.2 BodyPartPreference	54765 New section.	Y	New content added.
2.2.2.1	54395	N	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
E-mail Class Elements	Removed statement about the elements being children of the ApplicationData type.		update.
2.2.2.1.1 ConversationId	54395 Removed the reference link to [MS-ASCMD] and added a reference link to [MS-AEMAIL].	N	Content update.
2.2.2.1.1 ConversationId	54566 Added a statement to specify the condition in which the ConversationId element is not present. Removed content about null values.	N	Content update.
2.2.2.1.2 ConversationIndex	54395 Removed the reference to [MS-ASCMD] and added a reference to [MS-AEMAIL].	N	Content update.
2.2.2.1.2 ConversationIndex	54589 Clarified the conditions under which the delta-code bit (described in the first bulleted item) is set.	Y	Content update.
2.2.2.1.2 ConversationIndex	54590 Revised the description of the third bulleted item to state that the random number is generated by using an implementation-specific algorithm.	Y	Content update.
2.2.2.1.2 ConversationIndex	54755 Combined the descriptions of the two 4-bit parts (the last two bulleted items) into one description labeled "unused".	Y	Content update.
2.2.2.1.2 ConversationIndex	54811 Clarified how the first timestamp is stored in the FILETIME structure and stated that the low-order 3 bytes are set to zero.	Y	Content update.
2.2.2.1.2 ConversationIndex	55383 Added labels "delta-code", "time-delta", and "unused" to the bulleted items that describe the parts of the additional timestamps and clarified how the value of the time-delta (the second bulleted item) is determined.	Y	Content update.
2.2.2.3.2 Move.DstFldId	54531 Updated the requirement for the type of destination folder used.	N	Content update.
2.2.2.3.3 Move.Options.MoveAlways	54377 Changed wording from "... is to be moved always" to "... is to be always moved" to match the description in the table in section 2.2.2.	N	Content update.
2.2.2.3.4	54540	N	Content

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
Response.Move.Status	Revised the description of status value 6.		update.
2.2.2.3.4 Response.Move.Status	54531 Updated the description of status value 156.	N	Content update.
3.1.4.7 Finding a Conversation	54631 Removed the bulleted list of Store.Query settings.	N	Content update.
3.1.4.9 Applying a Conversation-based Filter	54380 Clarified how a conversation-based filter is used by the GetItemEstimate command.	N	Content update.
3.1.4.10 Requesting a Message Part	54765 New section.	Y	New content added.
3.2.5.2 Processing a GetItemEstimate Command	54579 Updated the status value 4 and its description.	N	Content update.
3.2.5.6 Filtering	54382 Clarified the definition of the aging-off process.	N	Content update.
3.2.5.6 Filtering	54697 Removed the paragraph about flags on conversations.	N	Content update.
3.2.5.6 Filtering	54698 Removed the paragraph about marking a conversation as read or unread.	N	Content update.
3.2.5.7 Sending a Message Part	54765 New section.	Y	New content added.

8 Index

A

[Applicability](#) 6

C

[Capability negotiation](#) 6

[Change tracking](#) 25

Client

[overview](#) 14

E

Examples

[overview](#) 21

F

[Fields - vendor-extensible](#) 6

G

[Glossary](#) 5

I

[Implementer – security considerations](#) 23

[Index of security parameters](#) 23

[Informative references](#) 6

[Introduction](#) 5

M

Messages

[transport](#) 7

N

[Normative references](#) 5

O

[Overview](#) 6

P

[Parameters – security index](#) 23

[Preconditions](#) 6

[Prerequisites](#) 6

[Product behavior](#) 24

R

References

[informative](#) 6

[normative](#) 5

[Relationship to other protocols](#) 6

S

Security

[implementer considerations](#) 23

[overview](#) 23

[parameter index](#) 23

Server

[overview](#) 17

[Standards assignments](#) 6

T

[Tracking changes](#) 25

[Transport](#) 7

V

[Vendor-extensible fields](#) 6

[Versioning](#) 6