

[MS-ASDTYPE]: ActiveSync Data Types

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 iplq@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
12/03/2008	1.0.0	Major	Initial Release.
03/04/2009	1.0.1	Editorial	Revised and edited technical content.
04/10/2009	2.0.0	Major	Updated technical content and applicable product releases.
07/15/2009	3.0.0	Major	Revised and edited for technical content.
11/04/2009	4.0.0	Major	Updated and revised the technical content.
02/10/2010	5.0.0	Major	Updated and revised the technical content.
05/05/2010	6.0.0	Major	Updated and revised the technical content.
08/04/2010	6.1	Minor	Clarified the meaning of the technical content.
11/03/2010	6.2	Minor	Clarified the meaning of the technical content.
03/18/2011	6.3	Minor	Clarified the meaning of the technical content.

Table of Contents

1	Introduction	4
1.1	Glossary	4
1.2	References.....	4
1.2.1	Normative References.....	4
1.2.2	Informative References	5
1.3	Overview	5
1.4	Relationship to Protocols and Other Structures	6
1.5	Applicability Statement.....	6
1.6	Versioning and Capability Negotiation.....	6
1.7	Vendor-Extensible Fields.....	6
2	Structure	7
2.1	boolean	7
2.2	container	7
2.3	dateTime	7
2.3.1	Time Zones and Daylight Saving Time	8
2.3.2	Calculating Dates and Times	9
2.4	enumeration	10
2.5	integer	10
2.6	string	10
2.6.1	Byte Array.....	10
2.6.2	E-Mail Address	10
2.6.3	Telephone Number	11
2.6.4	TimeZone.....	11
2.7	unsignedByte	11
3	Data Type Examples	12
3.1	boolean Examples.....	12
3.2	container Example	12
3.3	dateTime Examples.....	12
3.4	enumeration Example	12
3.5	integer Examples.....	13
3.6	string Examples.....	13
3.6.1	Byte Array Example	13
3.6.2	E-Mail Address Examples.....	13
3.6.3	Telephone Number Examples.....	13
3.6.4	TimeZone Example.....	13
3.7	unsignedByte Example	14
4	Security Considerations.....	15
5	Appendix A: Product Behavior	16
6	Change Tracking.....	17
7	Index	19

1 Introduction

This document specifies the required format of each data type used by the ActiveSync **XML schema definitions (XSDs)** definitions.

The ActiveSync protocol sends and receives data in Wide Area Protocol (WAP) Binary XML (WBXML) format. In order to ensure that both the client and the server have the same expectations about the format of the element data, the ActiveSync commands and **classes** use XSDs to define the data type of each element.

1.1 Glossary

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

class
Coordinated Universal Time (UTC)
Hypertext Transfer Protocol (HTTP)
Secure Sockets Layer (SSL)
Unicode
XML

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

base64 encoding
calendar
header
instance
meeting
Meeting object
organizer
XML schema
XML schema definition (XSD)

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.

[ISO-8601] International Organization for Standardization, "Data Elements and Interchange Formats - Information Interchange - Representation of Dates and Times", ISO/IEC 8601:2004, December 2004, <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=40874&ICS1=1&ICS2=140&ICS3=30>

Note There is a charge to download the specification.

[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>

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

[WBXML1.2] Martin, B., and Jano, B., Eds., "WAP Binary XML Content Format", June 1999, <http://www.w3.org/1999/06/NOTE-wbxml-19990624>

[XMLSCHEMA] World Wide Web Consortium, "XML Schema", September 2005, <http://www.w3.org/2001/XMLSchema>

[XMLSCHEMA2/2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes Second Edition", W3C Recommendation, October 2004, <http://www.w3.org/TR/xmlschema-2>

1.2.2 Informative References

[MS-ASAIRS] Microsoft Corporation, "[ActiveSync AirSyncBase Namespace Protocol Specification](#)", December 2008.

[MS-ASCAL] Microsoft Corporation, "[ActiveSync Calendar Class Protocol Specification](#)", December 2008.

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

[MS-ASCNTC] Microsoft Corporation, "[ActiveSync Contact Class Protocol Specification](#)", December 2008.

[MS-ASCON] Microsoft Corporation, "[ActiveSync Conversations Protocol Specification](#)", July 2009.

[MS-ASDOC] Microsoft Corporation, "[ActiveSync Document Class Protocol Specification](#)", December 2008.

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

[MS-ASMS] Microsoft Corporation, "[ActiveSync Short Message Service Protocol Specification](#)", July 2009.

[MS-ASNOTE] Microsoft Corporation, "[ActiveSync Notes Class Protocol Specification](#)", July 2009.

[MS-ASTASK] Microsoft Corporation, "[ActiveSync Tasks Class Protocol Specification](#)", December 2008.

[MS-GLOS] Microsoft Corporation, "[Windows Protocols Master Glossary](#)", March 2007.

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, <http://www.ietf.org/rfc/rfc2616.txt>

1.3 Overview

The data types specified in this document are divided into two categories:

- Types specified by the **XML schema** data types definition, as described in [\[XMLSCHEMA2/2\]](#). Most of the types specified by this document conform to this description.
- Structured string types, in which the value of the element is a string data type that conforms to a specified format. The **byte array** structure, as described section [2.6.1](#), is an example of a structured string type.

1.4 Relationship to Protocols and Other Structures

The data types are used by the ActiveSync content classes, as described in [\[MS-ASCAL\]](#), [\[MS-ASCNTC\]](#), [\[MS-ASCON\]](#), [\[MS-ASDOC\]](#), [\[MS-ASEMAIL\]](#), [\[MS-ASMS\]](#), [\[MS-ASNOTE\]](#), and [\[MS-ASTASK\]](#); by the AirSyncBase namespace, as described in [\[MS-ASAIRS\]](#); and by the commands described in [\[MS-ASCMD\]](#).

1.5 Applicability Statement

The data types specified in this document are applicable to all ActiveSync schemas.

1.6 Versioning and Capability Negotiation

None.

1.7 Vendor-Extensible Fields

None.

2 Structure

The following sections describe data types used by the ActiveSync protocols. All data sent by the ActiveSync protocol is text, but some of the text values adhere to the following text style data types, as specified by the schemas.

2.1 boolean

A **boolean** is an XML Schema primitive data type, as specified in [\[XMLSCHEMA2/2\]](#) section 3.2.2. It is declared in XML Schema syntax as an <element> with a **type** attribute of "boolean".

The value of a **boolean** element is an integer whose only valid values are 1 (TRUE) or 0 (FALSE). If the integer value is missing, then it is assumed to be 1 (TRUE). For examples, see section [3.1](#). Elements with a **boolean** data type MUST be encoded and transmitted as [\[WBXML1.2\]](#) inline strings.

2.2 container

A **container** is an **XML** element that encloses other elements but has no value of its own. It is a complex type with complex content, as specified in [\[XMLSCHEMA\]](#) section 3.4.2. It is defined in XML Schema syntax using a <complexType> element that specifies the allowable children for that element using the XML Schema <element> tag.

2.3 dateTime

A **dateTime** is a primitive XML Schema data type, as specified in [\[XMLSCHEMA2/2\]](#) section 3.2.7. It is declared in XML Schema syntax as an <element> whose **type** attribute is set to "dateTime".

dateTime values are as specified in [\[ISO-8601\]](#).

All dates are given in **Coordinated Universal Time (UTC)** and are represented as a string in the following format.

YYYY-MM-DDTHH:MM:SS.MSSZ where

YYYY = Year (Gregorian **calendar** year)

MM = Month (01 - 12)

DD = Day (01 - 31)

HH = Number of complete hours since midnight (00 - 24)

MM = Number of complete minutes since start of hour (00 - 59)

SS = Number of seconds since start of minute (00 - 59)

MSS = Number of milliseconds

The T serves as a separator, and the Z indicates that this time is in UTC.

For example, 8:35 A.M. on December 25, 2000 would be represented as 2000-12-25T08:35:00.000Z.

Note Dates and times in calendar items MUST NOT include punctuation separators. For example:
<A:StartTime>20021126T160000Z</A:StartTime>

Elements with a **dateTime** data type MUST be encoded and transmitted as [\[WBXML1.2\]](#) inline strings.

2.3.1 Time Zones and Daylight Saving Time

Dates and times can be very simple in calendars that are not shared. All times MAY be in device-local time, and there is no need for time zones or Daylight Savings Time (DST). If a **meeting** is scheduled for 10:00 A.M., it is in device time and, if the user of the device travels to another time zone, he or she adjusts the device time, but the meeting time remains at 10:00 A.M. If DST begins, the device time is adjusted again, but the meeting time remains at 10:00 A.M.

Dates and times become more complex when calendar events are shared by people who are in different time zones and are not all on DST. If Sean in Seattle schedules a 10:00 A.M. conference call with Annette in New York, the meeting will appear at 1:00 P.M. on Annette's calendar. If Jeff in Arizona is also on the call, he should see the meeting in his local time on his calendar. Because Arizona does not observe DST, the meeting is shown at 11:00 A.M. if it is the winter, but at 10:00 A.M. if it is the summer. If the meeting is recurring, then the dates and times are more complex during the transitions between DST and standard time. The following table lists the local and UTC times for a 10:00 A.M. meeting the weeks before and after the transition to DST.

Date	Seattle	Arizona	New York	UTC
4/4/03	10:00 Pacific Time (PT)	11:00 MST (Mountain Standard Time)	13:00 Eastern Standard Time (EST)	18:00 UTC
4/11/03	10:00 Pacific Daylight Time (PDT)	10:00 MST	13:00 Eastern Daylight Time (EDT)	17:00 UTC

The Seattle time remains the same before and after the transition to DST because the meeting **organizer** is in Seattle. If the organizer was Jeff in Arizona, then the meeting times before and after the DST transition would be different, as shown in the following table.

Date	Seattle	Arizona	New York	UTC
4/4/03	10:00 PT	11:00 MST	13:00 EST	18:00 UTC
4/11/03	11:00 PDT	11:00 MST	14:00 EDT	18:00 UTC

The shared **Meeting object** in the calendar application MUST store the following information. For a one-time meeting, the UTC time alone MAY be stored, and each device MAY translate to its local time by using its local time zone information. The time zone information includes a permanent time zone offset and, if appropriate, DST start and end dates, and time bias.

If the meeting is recurring, however, the UTC time can change depending on whether daylight saving time is in effect at the originator's location for each occurrence. The constant is the time in the originator's time zone, which is the time that MUST be stored. In addition, the originator's time zone MUST be stored. To display a meeting time, the time MUST be converted to UTC by using the originator's time zone, and then it MUST be converted to local time by using the device's local time zone.

Note The UTC time MAY be stored instead of the originator's local time. But the originator's time zone MUST still be stored. This feature allows for the daylight saving time adjustment, although the calculation is somewhat less intuitive.

If this recurring meeting has an exception, then the exception contains the date and time of the series **instance** that is different. As with the series itself, the UTC of the exception varies based on

DST. Therefore, the originator's time zone **MUST** be used to calculate the time of the exception. Because the originator's time zone is stored with the recurrence, the time zone **MAY NOT** be stored again for each exception.

2.3.2 Calculating Dates and Times

The ActiveSync protocols use the UTC time and the originator's time zone for all meetings. For single occurrences, the device converts the time to the local time zone. The originator's time zone is not important because the original conversion to UTC accounts for time zone and Daylight Savings Time (DST). However, for recurring meetings, the device **MUST** consider the possibility of a transition into or out of DST during the series. The stored UTC corresponds to the first occurrence of the series, but later meetings can have different corresponding UTC times. Therefore, to display the correct time, the device **MUST** perform one calculation that accounts for the originator's time zone, in addition to the device's local time zone.

The following table shows the time zone information for the earlier examples.

	Pacific Time	Mountain Time (Arizona)	Eastern Time
Time zone offset	UTC-8	UTC-7	UTC-5
Daylight start	4/6/03 02:00	None	4/6/03 02:00
Daylight end	10/26/03 02:00	None	10/26/03 02:00
Daylight bias	+1	0	+1

The calculation to display the local time of a meeting instance is as follows:

(Meeting time in UTC) + (local time zone offset) + (local daylight bias) – (original daylight bias)

Note Daylight bias is a time zone's offset during DST. The local daylight bias comes from the local time zone information, and the original daylight bias comes from the originator's time zone information.

The weekly conference call repeats every Friday beginning 4/4/03. The start time of the first instance is 10:00 A.M. PT, or 18:00 UTC. Therefore, the stored time is 18:00 and the time zone is Pacific Time.

Date	Seattle	Arizona	New York
4/4/03	$1800 + (-8) + (0) - (0) = 1000$	$1800 + (-7) + (0) - (0) = 1100$	$1800 + (-5) + (0) - (0) = 1300$
4/11/03	$1800 + (-8) + (+1) - (+1) = 1000$	$1800 + (-7) + (0) - (+1) = 1000$	$1800 + (-5) + (+1) - (+1) = 1300$

Notice that both the local and original DST biases are the ones in effect on the date/time of the meeting instance.

The weekly conference call repeats every Friday beginning on 4/4/03. The originator was in Arizona, so the start time of the first instance is 11:00 MST (Arizona), or 18:00 UTC. The stored time is 18:00 and the time zone is MST (Arizona).

Date	Seattle	Arizona	New York
4/4/03	$1800 + (-8) + (0) - (0) = 1000$	$1800 + (-7) + (0) - (0) = 1100$	$1800 + (-5) + (0) - (0) = 1300$

Date	Seattle	Arizona	New York
4/11/03	$1800+(-8)+(+1)-(0) = 1100$	$1800+(-7)+(0)-(0) = 1100$	$1800+(-5)+(+1)-(0) = 1400$

2.4 enumeration

An **enumeration** specifies a fixed set of values for an element or attribute. In accordance with [\[XMLSCHEMA2/2\]](#) section 4.3.5, it is specified using the <restriction> element to declare the enumeration, and the <enumeration> element to define one or more allowed values.

2.5 integer

An **integer** is a numeric value that can be provided in the XML body of a command or in a **POST** command **header**. It is an XML Schema primitive data type, as specified in [\[XMLSCHEMA2/2\]](#) section 3.3.13. Elements with an **integer** data type MUST be encoded and transmitted as [\[WBXML1.2\]](#) inline strings.

2.6 string

A **string** is a chunk of **Unicode** text. It is an XML Schema primitive data type as specified in [\[XMLSCHEMA2/2\]](#) section 3.2.1. An element of this type is declared in XML Schema as an <element> with a **type** attribute of "string".

Elements with a **string** data type MUST be encoded and transmitted as [\[WBXML1.2\]](#) inline strings.

Some **string** values are constrained to a particular set of values, which is included in the description of the element.

ActiveSync defines several conventions for strings that adhere to commonly used formats:

- **Byte Array** (section [2.6.1](#))
- **E-mail Address** (section [2.6.2](#))
- **Telephone Number** (section [2.6.3](#))
- **TimeZone** (section [2.6.4](#))

Elements of these types are defined as **string** types in XML Schema, but commands that process such elements can return an error if the value of the element does not adhere to the expected format.

2.6.1 Byte Array

A **byte array** is a structure inside of an element of the **string** type (section [2.6](#)). The structure is comprised of a length, which is expressed as a multi-byte integer (as specified in [\[WBXML1.2\]](#)), followed by that many bytes of data. Elements with a **byte array** structure MUST be encoded and transmitted as [\[WBXML1.2\]](#) opaque data.

2.6.2 E-Mail Address

An e-mail address is an unconstrained value of an element of the **string** type (section [2.6](#)).

However, a valid individual e-mail address MUST have the following format: "local-part@domain". For more information about e-mail address syntax, see [\[RFC822\]](#) section 6.

2.6.3 Telephone Number

A telephone number is an unconstrained value of elements of the **string** type (section [2.6](#)) that can include an area code and a country code.

2.6.4 TimeZone

The **TimeZone** structure is a time zone structure encoded with **base64 encoding** with the following definition:

```
typedef struct {
    LONG Bias;
    WCHAR StandardName[32];
    SYSTEMTIME StandardDate;
    LONG StandardBias;
    WCHAR DaylightName[32];
    SYSTEMTIME DaylightDate;
    LONG DaylightBias;
};
```

The required values are **Bias**, which is the offset from UTC, in minutes, and the **StandardDate** and **DaylightDate**, which are needed when the biases take effect. For example, the bias for Pacific Time is 480.

The **TimeZone** structure is included in elements whose values are defined by the **string** type (section [2.6](#)). Elements with a **TimeZone** structure MUST be encoded and transmitted as [\[WBXML1.2\]](#) inline strings.

2.7 unsignedByte

The **unsignedByte** data type is an integer value between 0 and 255, inclusive. It is an XML Schema primitive data type as specified in [\[XMLSCHEMA2/2\]](#) section 3.3.24. Elements of this type are declared in XML Schema syntax with an `<element>` whose **type** attribute is set to "unsignedByte".

3 Data Type Examples

3.1 boolean Examples

Note in the following examples that the short form `<Tag />` is equivalent to `<Tag>1</Tag>`.

```
<email:Read>0</email:Read>
<email:AllDayEvent>1</email:AllDayEvent>
<email:AllDayEvent />
```

3.2 container Example

In the following example, `<FolderCreate>` is a container.

```
<?xml version="1.0" encoding="utf-8"?>
<FolderCreate xmlns="FolderHierarchy:">
  <FolderCreate>
    <ServerId>1</ServerId>
    <ParentId>0</ParentId>
    <DisplayName>Calendar</DisplayName>
    <Type>8</Type>
  </FolderCreate>
```

3.3 dateTime Examples

The following example demonstrates the **dateTime** format as used by the Email class, as described in [\[MS-ASEMAIL\]](#).

```
<?xml version="1.0" encoding="utf-8"?>
<Sync xmlns:email="Email:" xmlns:airsyncbase="AirSyncBase:" xmlns:email2="Email2:"
  xmlns="AirSync:">

  <A:DateReceived>2009-11-12T00:45:06.000Z</A:DateReceived>
```

The following example demonstrates the **dateTime** format used by the Calendar class, as described in [\[MS-ASCAL\]](#).

```
<?xml version="1.0" encoding="utf-8"?>
<Sync xmlns="AirSync:" xmlns:calendar="Calendar:" xmlns:airsyncbase="AirSyncBase:">

  ...

  <airsyncbase:StartTime>20091212T000000Z</airsyncbase:StartTime>

  ...
```

3.4 enumeration Example

The allowed **enumeration** values are defined in the schema.

```

<xs:element name="UserResponse">
  <xs:simpleType>
    <xs:restriction base="xs:unsignedByte">
      <xs:enumeration value="3"/>
      <xs:enumeration value="1"/>
      <xs:enumeration value="2"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

3.5 integer Examples

```

<airsynbase:TruncationSize>456</airsynbase:TruncationSize>
<airsync:FilterType>3</airsync:FilterType>
<airsync:Status>1</airsync:Status>

```

3.6 string Examples

```

<contact:CompanyName>Adventure Works</contact:CompanyName>
<contact:BusinessPhoneNumber>(800) 555-0100</contact:BusinessPhoneNumber>
<email:MessageClass>IPM.NOTE</email:MessageClass>

```

3.6.1 Byte Array Example

In this example, the continuation flag (as described in [WBXML1.2](#)) is not set, indicating that the length is only one byte long. This results in a length of 4 bytes. The following 4 bytes compromise the data.

```
04 00 01 02 03
```

3.6.2 E-Mail Address Examples

```

<resolverecipients:Recipient>amy@nowhere.com</resolverecipients:Recipient>
<email2:Sender>j.smith@nowhere.com</email2:Sender>

```

3.6.3 Telephone Number Examples

```

<contacts:HomePhoneNumber>3605551212</contacts:HomePhoneNumber>
<contacts:BusinessPhoneNumber>+011 (73) 5551212</contacts:BusinessPhoneNumber>

```

3.6.4 TimeZone Example

```

<email:TimeZone>
4AEAACgARwBNAFQALQAwADgAOGAwADAACKQAgAFAAYQBjAGkAZgBpAGMAIABUAGkAbQB1ACAAKABVA
FMAIAAmACAAQwAAAAaAAABAAIAAAAAAAAAAAAAACgARwBNAFQALQAwADgAOGAwADAACKQAgAFAAYQ
BjAGkAZgBpAGMAIABUAGkAbQB1ACAAKABVAFMAIAAmACAAQwAAAAAMAAACAAIAAAAAAAAAAxP//w=
=
</email:TimeZone>

```

3.7 unsignedByte Example

```
<calendar:BusyStatus>3</calendar:BusyStatus>
```

4 Security Considerations

In most cases, all communication between the client and server happens across an **HTTP** connection secured by the **Secure Sockets Layer (SSL)** protocol, as described in [\[RFC2616\]](#). The SSL connection is assumed to be secure enough to transmit confidential data, such as user credentials and sensitive e-mail. The SSL certificate on the server is assumed to be trusted by the client application.

5 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 2007 Service Pack 1 (SP1)
- Microsoft® Exchange Server 2010

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.

6 Change Tracking

This section identifies changes that were made to the [MS-ASDTYPE] protocol document between the November 2010 and March 2011 releases. 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.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 language and 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 or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- 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 protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.1 Glossary	Removed "ASCII" and "exception" from list of terms.	N	Content updated.
1.3 Overview	Added reference to [XMLSCHEMA2/2].	N	Content updated.

7 Index

A

[Applicability](#) 6

C

[Capability negotiation](#) 6

[Change tracking](#) 17

F

[Fields - vendor-extensible](#) 6

G

[Glossary](#) 4

I

[Implementer - security considerations](#) 15

[Informative references](#) 5

[Introduction](#) 4

N

[Normative references](#) 4

O

[Overview](#) 5

P

[Product behavior](#) 16

R

References

[informative](#) 5

[normative](#) 4

[Relationship to protocols and other structures](#) 6

S

[Security - implementer considerations](#) 15

T

[Tracking changes](#) 17

V

[Vendor-extensible fields](#) 6

[Versioning](#) 6