Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **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 can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks.** The names of companies and products contained in this documentation might 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, email addresses, logos, people, places, and events that are 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 as specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications documentation does 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 documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact dochelp@microsoft.com.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/15/2009</td>
<td>1.0</td>
<td>Major</td>
<td>Initial Availability.</td>
</tr>
<tr>
<td>11/4/2009</td>
<td>1.0.1</td>
<td>Editorial</td>
<td>Revised and edited the technical content.</td>
</tr>
<tr>
<td>2/10/2010</td>
<td>2.0.0</td>
<td>Major</td>
<td>Updated and revised the technical content.</td>
</tr>
<tr>
<td>5/5/2010</td>
<td>3.0.0</td>
<td>Major</td>
<td>Updated and revised the technical content.</td>
</tr>
<tr>
<td>8/4/2010</td>
<td>3.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>11/3/2010</td>
<td>4.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>3/18/2011</td>
<td>5.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>8/5/2011</td>
<td>5.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>10/7/2011</td>
<td>5.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>1/20/2012</td>
<td>6.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/27/2012</td>
<td>6.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>7.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>7.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>7.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/26/2013</td>
<td>7.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>8.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>8.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>9.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>9.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>10.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>5/26/2015</td>
<td>11.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/14/2015</td>
<td>12.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>6/13/2016</td>
<td>13.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>13.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/15/2016</td>
<td>13.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>9/19/2017</td>
<td>13.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/24/2018</td>
<td>14.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>10/1/2018</td>
<td>15.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>11/19/2019</td>
<td>15.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/15/2022</td>
<td>15.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction .............................................................................. 6
  1.1 Glossary ............................................................................. 6
  1.2 References .......................................................................... 7
    1.2.1 Normative References ......................................................... 8
    1.2.2 Informative References ....................................................... 9
  1.3 Overview .............................................................................. 9
  1.4 Relationship to Other Protocols ............................................. 9
  1.5 Prerequisites/Preconditions .................................................. 9
  1.6 Applicability Statement ....................................................... 9
  1.7 Versioning and Capability Negotiation .................................. 9
  1.8 Vendor-Extensible Fields ..................................................... 10
  1.9 Standards Assignments ....................................................... 10

2 Messages.................................................................................. 11
  2.1 Transport ............................................................................ 11
  2.2 Common Message Syntax ................................................... 11
    2.2.1 Namespaces ...................................................................... 11
    2.2.2 Messages ........................................................................ 11
    2.2.3 Elements ....................................................................... 11
    2.2.4 Complex Types .............................................................. 12
      2.2.4.1 t:ArrayOfAttachmentsType Complex Type ....................... 12
      2.2.4.2 t:AttachmentIdType Complex Type .............................. 13
      2.2.4.3 m:AttachmentInfoResponseMessageType Complex Type 13
      2.2.4.4 t:AttachmentType Complex Type ................................. 14
      2.2.4.5 t:FileAttachmentType Complex Type ............................ 15
      2.2.4.6 t:ItemAttachmentType Complex Type ......................... 16
      2.2.4.7 t:NonEmptyArrayOfRequestAttachmentIdsType Complex Type 18
      2.2.4.8 t:ReferenceAttachmentType Complex Type ................... 18
      2.2.4.9 t:RequestAttachmentIdType Complex Type .................. 19
    2.2.5 Simple Types .................................................................. 20
    2.2.6 Attributes ..................................................................... 20
    2.2.7 Groups .......................................................................... 20
    2.2.8 Attribute Groups ........................................................... 20

3 Protocol Details ....................................................................... 21
  3.1 ExchangeServicePortType Server Details ............................... 21
    3.1.1 Abstract Data Model ....................................................... 21
    3.1.2 Timers ........................................................................... 21
    3.1.3 Initialization ................................................................... 21
    3.1.4 Message Processing Events and Sequencing Rules ............... 21
    3.1.4.1 CreateAttachment Operation ......................................... 21
      3.1.4.1.1 Messages .................................................................. 22
      3.1.4.1.2 Elements .................................................................. 23
      3.1.4.1.2.1 CreateAttachment Element ................................... 24
      3.1.4.1.2.2 CreateAttachmentResponse Element .................... 24
      3.1.4.1.3 Complex Types ....................................................... 24
      3.1.4.1.3.1 m:CreateAttachmentResponseType Complex Type .... 24
      3.1.4.1.3.2 m:CreateAttachmentType Complex Type .............. 25
    3.1.4.2 DeleteAttachment Operation ......................................... 25
      3.1.4.2.1 Messages .................................................................. 26
      3.1.4.2.1.1 tns:DeleteAttachmentSoapIn Message ..................... 26
      3.1.4.2.1.2 tns:DeleteAttachmentSoapOut Message .................. 27
      3.1.4.2.2 Elements .................................................................. 27
3.1.4.2.2.2 DeleteAttachmentResponse Element ..................................................... 28
3.1.4.2.3 Complex Types ......................................................................................... 28
3.1.4.2.3.1 m:DeleteAttachmentResponseMessageType Complex Type .............. 28
3.1.4.2.3.2 m:DeleteAttachmentResponseType Complex Type ......................... 29
3.1.4.2.3.3 m:DeleteAttachmentType Complex Type ............................................. 29
3.1.4.2.3.4 t:RootItemIdType Complex Type ......................................................... 30
3.1.4.3 GetAttachment Operation ........................................................................... 30
3.1.4.3.1 Messages ................................................................................................. 31
3.1.4.3.1.1 tns:GetAttachmentSoapIn Message ..................................................... 31
3.1.4.3.1.2 tns:GetAttachmentSoapOut Message .................................................. 32
3.1.4.3.2 Elements ................................................................................................. 32
3.1.4.3.2.1 GetAttachment Element .................................................................... 33
3.1.4.3.2.2 GetAttachmentResponse Element .................................................... 33
3.1.4.3.3 Complex Types ......................................................................................... 33
3.1.4.3.3.1 m:GetAttachmentResponseType Complex Type ............................. 33
3.1.4.3.3.2 m:GetAttachmentType Complex Type ................................................. 33
3.1.4.3.3.3 t:AttachmentResponseShapeType Complex Type ............................. 34
3.1.5 Timer Events ................................................................................................. 35
3.1.6 Other Local Events ....................................................................................... 35
3.2 Client Details .................................................................................................. 35
3.2.1 Abstract Data Model ..................................................................................... 35
3.2.2 Timers ........................................................................................................... 35
3.2.3 Initialization .................................................................................................. 35
3.2.4 Message Processing Events and Sequencing Rules ....................................... 35
3.2.5 Timer Events ................................................................................................. 36
3.2.6 Other Local Events ....................................................................................... 36

4 Protocol Examples ................................................................................................ 37
4.1 CreateAttachment Example ............................................................................ 37
4.2 DeleteAttachment Example ............................................................................ 38
4.3 GetAttachment Example ................................................................................ 39

5 Security .............................................................................................................. 41
5.1 Security Considerations for Implementers ....................................................... 41
5.2 Index of Security Parameters ............................................................................ 41

6 Appendix A: Full WSDL .................................................................................... 42

7 Appendix B: Full XML Schema .......................................................................... 45
7.1 Messages Schema ............................................................................................ 45
7.2 Types Schema .................................................................................................. 46

8 Appendix C: Product Behavior ........................................................................... 49

9 Change Tracking ................................................................................................ 50

10 Index .................................................................................................................. 51
1 Introduction

The Attachment Handling Web Service Protocol is used to create, delete, and get attachments on items on the server.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

- **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.

- **base64 encoding**: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

- **endpoint**: A communication port that is exposed by an application server for a specific shared service and to which messages can be addressed.

- **Hypertext Transfer Protocol (HTTP)**: An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

- **Hypertext Transfer Protocol Secure (HTTPS)**: An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [SSL3] and [RFC5246].

- **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.

- **Multipurpose Internet Mail Extensions (MIME)**: A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], [RFC2046], and [RFC2047].

- **Object Linking and Embedding (OLE)**: A technology for transferring and sharing information between applications by inserting a file or part of a file into a compound document. The inserted file can be either embedded or linked. See also embedded object and linked object.

- **SOAP**: A lightweight protocol for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics. SOAP 1.2 supersedes SOAP 1.1. See [SOAP1.2-1/2003].

- **SOAP body**: A container for the payload data being delivered by a SOAP message to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.
SOAP header: A mechanism for implementing extensions to a SOAP message in a decentralized manner without prior agreement between the communicating parties. See [SOAP1.2-1/2007] section 5.2 for more information.

SOAP message: An XML document consisting of a mandatory SOAP envelope, an optional SOAP header, and a mandatory SOAP body. See [SOAP1.2-1/2007] section 5 for more information.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].

Web Services Description Language (WSDL): An XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are described abstractly and are bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints, which describe a network service. WSDL is extensible, which allows the description of endpoints and their messages regardless of the message formats or network protocols that are used.

WSDL message: An abstract, typed definition of the data that is communicated during a WSDL operation [WSDL]. Also, an element that describes the data being exchanged between web service providers and clients.

WSDL operation: A single action or function of a web service. The execution of a WSDL operation typically requires the exchange of messages between the service requestor and the service provider.

WSDL port type: A named set of logically-related, abstract Web Services Description Language (WSDL) operations and messages.

XML: The Extensible Markup Language, as described in [XML1.0].

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML schema: A description of a type of XML document that is typically expressed in terms of constraints on the structure and content of documents of that type, in addition to the basic syntax constraints that are imposed by XML itself. An XML schema provides a view of a document type at a relatively high level of abstraction.

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

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.
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-OXWSCDATA] Microsoft Corporation, "Common Web Service Data Types".

[MS-OXWSCONT] Microsoft Corporation, "Contacts Web Service Protocol".

[MS-OXWSSCORE] Microsoft Corporation, "Core Items Web Service Protocol".

[MS-OXWSGTZ] Microsoft Corporation, "Get Server Time Zone Web Service Protocol".

[MS-OXWSMSG] Microsoft Corporation, "Email Message Types Web Service Protocol".

[MS-OXWSMTGS] Microsoft Corporation, "Calendaring Web Service Protocol".

[MS-OXWSPOST] Microsoft Corporation, "Post Items Web Service Protocol".

[MS-OXWSTASK] Microsoft Corporation, "Tasks Web Service Protocol".


1.2.2 Informative References


1.3 Overview

The Attachment Handling Web Service Protocol enables clients to create, get, and delete attachments on messages. Clients use Attachment objects to associate files, Object Linking and Embedding (OLE) objects, other messages, or binary data with a particular Message object. Because Attachment objects are created, maintained, and accessed only in the context of a message, they are considered to be subobjects.

1.4 Relationship to Other Protocols

A client that implements this protocol can use the Autodiscover Publishing and Lookup SOAP-Based Web Service Protocol [MS-OXWSADISC] or the Autodiscover Publishing and Lookup Protocol [MS-OXDSCLI] to identify the target endpoint to use for each operation.

This protocol uses the SOAP protocol as described in [SOAP1.1] to specify the structure information exchanged between the client and server. This protocol uses the XML protocol as described in [XMLSCHEMA1] and [XMLSCHEMA2] to describe the message content sent to and from the server.

This protocol uses SOAP over HTTP as described in [RFC2616], and SOAP over HTTPS as described in [RFC2818], as shown in the following figure.

Figure 1: This protocol in relation to other protocols

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

1.5 Prerequisites/Preconditions

The endpoint URL that is returned by either the Autodiscover Publishing and Lookup SOAP-Based Web Service Protocol, as described in [MS-OXWSADISC], or the Autodiscover Publishing and Lookup Protocol, as described in [MS-OXDSCLI], is required to form the HTTP request to the Web server that hosts this protocol. The operations that this protocol defines cannot be accessed unless the correct endpoint is identified in the HTTP Web requests that target this protocol.

1.6 Applicability Statement

The protocol specified in this document is applicable to environments that create, get, and delete attachments on messages. This Web service protocol is applicable to SOAP-based clients [SOAP1.1].

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:
- **Supported Transports:** This protocol uses multiple transports with SOAP 1.1, as specified in section 2.1.

- **Protocol Versions:** This protocol has only one WSDL port type version. The WSDL version of the request is identified using the `t:RequestServerVersion` element as described in [MS-OXWSCDATA] section 2.2.3.9, and the version of the server responding to the request is identified using the `t:ServerVersionInfo` element as described in [MS-OXWSCDATA] section 2.2.3.10.

- **Security and Authentication Methods:** This protocol relies on the Web server that is hosting it to perform authentication.

- **Localization:** This protocol includes text strings in various messages. Localization considerations for such strings are specified in sections 2.2 and 3.1.4.

- **Capability Negotiation:** This protocol does not support version negotiation.

### 1.8 Vendor-Extensible Fields

None.

### 1.9 Standards Assignments

None.
2 Messages

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The WSDL in this specification provides a base description of the protocol. The schema in this specification provides a base description of the message syntax. The text that specifies the WSDL and schema might specify restrictions that reflect actual protocol behavior. For example, the schema definition might allow for an element to be empty, null, or not present but the behavior of the protocol as specified restricts the same elements to being non-empty, not null, or present.

2.1 Transport

The SOAP version supported is SOAP 1.1. For details, see [SOAP1.1].

This protocol relies on the Web server that hosts the application to perform authentication. This protocol MUST support SOAP over HTTP, as specified in [RFC2616]. The protocol SHOULD use secure communications via HTTPS, as defined in [RFC2818].

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses XML schema, as defined in [XMLSCHEMA1] and [XMLSCHEMA2], and Web Services Description Language (WSDL), as defined in [WSDL].

2.2.1 Namespaces

This specification defines and references various XML namespaces by using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace URI</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>soap</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/soap/">http://schemas.xmlsoap.org/wsdl/soap/</a></td>
<td>[SOAP1.1]</td>
</tr>
<tr>
<td>tns</td>
<td><a href="http://schemas.microsoft.com/exchange/services/2006/messages">http://schemas.microsoft.com/exchange/services/2006/messages</a></td>
<td></td>
</tr>
<tr>
<td>xs</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>[XMLSCHEMA1]</td>
</tr>
<tr>
<td>wsd1</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a></td>
<td>[WSDL]</td>
</tr>
<tr>
<td>wsi</td>
<td><a href="http://ws-i.org/schemas/conformanceClaim/">http://ws-i.org/schemas/conformanceClaim/</a></td>
<td>[WSIBASIC]</td>
</tr>
<tr>
<td>t</td>
<td><a href="http://schemas.microsoft.com/exchange/services/2006/types">http://schemas.microsoft.com/exchange/services/2006/types</a></td>
<td></td>
</tr>
<tr>
<td>m</td>
<td><a href="http://schemas.microsoft.com/exchange/services/2006/messages">http://schemas.microsoft.com/exchange/services/2006/messages</a></td>
<td></td>
</tr>
</tbody>
</table>

2.2.2 Messages

This specification does not define any common WSDL message definitions.

2.2.3 Elements

This specification does not define any common XML schema element definitions.
2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions that are defined by this specification. XML schema complex type definitions that are specific to a particular operation are described with the operation.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayOfAttachmentsType</td>
<td>Represents an array of types based on attachments on the item.</td>
</tr>
<tr>
<td>AttachmentIdType</td>
<td>Represents the item identifier and new change key of an item after an attachment has been attached.</td>
</tr>
<tr>
<td>AttachmentInfoResponseMessageType</td>
<td>Contains status and response data for attachments.</td>
</tr>
<tr>
<td>AttachmentType</td>
<td>Represents an attachment.</td>
</tr>
<tr>
<td>FileAttachmentType</td>
<td>Represents a file that is attached to an item in the server store.</td>
</tr>
<tr>
<td>ItemAttachmentType</td>
<td>Represents an item that is attached to another item.</td>
</tr>
<tr>
<td>NonEmptyArrayOfRequestAttachmentIdsType</td>
<td>Represents an array of attachment identifiers.</td>
</tr>
<tr>
<td>ReferenceAttachmentType</td>
<td>Represents a reference that is attached another item.</td>
</tr>
<tr>
<td>RequestAttachmentIdType</td>
<td>Represents the identifier for an attachment.</td>
</tr>
</tbody>
</table>

2.2.4.1 t:ArrayOfAttachmentsType Complex Type

The ArrayOfAttachmentsType complex type represents an array of types based on attachments on the item. The ArrayOfAttachmentsType complex type is used only in the response message.

```xml
<xs:complexType name="ArrayOfAttachmentsType">
  <xs:choice
    minOccurs="0"
    maxOccurs="unbounded">
    <xs:element name="ItemAttachment"
      type="t:ItemAttachmentType"/>
    <xs:element name="FileAttachment"
      type="t:FileAttachmentType"/>
    <xs:element name="ReferenceAttachment"
      type="t:ReferenceAttachmentType"/>
  </xs:choice>
</xs:complexType>
```

The following table lists the child elements of the ArrayOfAttachmentsType complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItemAttachment</td>
<td>t:ItemAttachmentType (section 2.2.4.6)</td>
<td>Specifies an item that is attached to another item.</td>
</tr>
<tr>
<td>FileAttachment</td>
<td>t:FileAttachmentType (section 2.2.4.5)</td>
<td>Specifies a file that is attached to</td>
</tr>
</tbody>
</table>
2.2.4.2  t:AttachmentIdType Complex Type

The AttachmentIdType complex type represents the item identifier and new change key of an item after an attachment has been attached. The AttachmentIdType complex type extends the RequestAttachmentIdType complex type, as specified in section 2.2.4.9.

```xml
<xs:complexType name="AttachmentIdType">
    <xs:complexContent>
        <xs:extension base="t:RequestAttachmentIdType">
            <xs:attribute name="RootItemId" type="xs:string" use="optional"/>
            <xs:attribute name="RootItemChangeKey" type="xs:string" use="optional"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table summarizes the set of common XML schema attribute definitions defined by this specification.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RootItemId</td>
<td>xs:string</td>
<td>Represents the unique identifier of the root store item to which the attachment is attached. The store item is an item in the server store. The maximum length is 512 bytes after base64 decoding.</td>
</tr>
<tr>
<td>RootItemChangeKey</td>
<td>xs:string</td>
<td>Represents the change key of the root store item to which the attachment is attached. The maximum length is 512 bytes after base64 decoding.</td>
</tr>
</tbody>
</table>

2.2.4.3  m:AttachmentInfoResponseMessageType Complex Type

The AttachmentInfoResponseMessageType complex type contains status and response data for attachments. The AttachmentInfoResponseMessageType complex type extends the ResponseMessageType complex type, ([MS-OXWSCDATA] section 2.2.4.67).

```xml
<xs:complexType name="AttachmentInfoResponseMessageType">
    <xs:complexContent>
        <xs:extension base="m:ResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
```
The following table lists the child element of the **AttachmentInfoResponseMessageType** complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attachments</td>
<td>t:ArrayOfAttachmentsType</td>
<td>Represents an array of types based on attachments on the item. The <strong>ArrayOfAttachmentsType</strong> complex type is used only in the response message.</td>
</tr>
</tbody>
</table>

### 2.2.4.4 t:AttachmentType Complex Type

The **AttachmentType** complex type represents an attachment.

```xml
<xs:complexType name="AttachmentType">
  <xs:sequence>
    <xs:element name="AttachmentId" type="t:AttachmentIdType" minOccurs="0" maxOccurs="1" />
    <xs:element name="Name" type="xs:string" minOccurs="0" maxOccurs="1" />
    <xs:element name="ContentType" type="xs:string" minOccurs="0" maxOccurs="1" />
    <xs:element name="ContentId" type="xs:string" minOccurs="0" maxOccurs="1" />
    <xs:element name="ContentLocation" type="xs:string" minOccurs="0" maxOccurs="1" />
    <xs:element name="Size" type="xs:int" minOccurs="0" maxOccurs="1" />
    <xs:element name="LastModifiedTime" type="xs:dateTime" minOccurs="0" maxOccurs="1" />
    <xs:element name="IsInline" type="xs:boolean" minOccurs="0" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>
```
The following table lists the child elements of the **AttachmentType** complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentId</td>
<td>t:AttachmentIdType</td>
<td>Specifies the attachment identifier.</td>
</tr>
<tr>
<td></td>
<td>(section 2.2.4.2)</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>xs:string ([XMLSCHEMA2])</td>
<td>Specifies the MIME type of the attachment content. For example: &quot;text/enriched&quot;, &quot;text/html&quot;, &quot;text/plain&quot;, &quot;text/rfc822-headers&quot;, &quot;text/richtext&quot;, or &quot;text/sgml&quot;.</td>
</tr>
<tr>
<td>ContentType</td>
<td>xs:string</td>
<td>Specifies the <strong>MIME</strong> type of the attachment content.</td>
</tr>
<tr>
<td>ContentId</td>
<td>xs:string</td>
<td>Specifies the unique object identifier for an attachment.</td>
</tr>
<tr>
<td>ContentLocation</td>
<td>xs:string</td>
<td>Specifies the <strong>URI</strong> that corresponds to the location of the content of the attachment. The <strong>ContentLocation</strong> element can be used to associate an attachment with a URL that defines its location on the Web.</td>
</tr>
<tr>
<td>Size</td>
<td>xs:int ([XMLSCHEMA2])</td>
<td>Specifies an estimate of the size, in bytes, of the item's complete body.</td>
</tr>
<tr>
<td>LastModifiedTime</td>
<td>xs:dateTime ([XMLSCHEMA2])</td>
<td>Specifies the day and time that this item was last changed.</td>
</tr>
<tr>
<td>IsInline</td>
<td>&lt;1&gt;</td>
<td>A Boolean value that indicates whether the attachment is an inline attachment. The <strong>IsInline</strong> element is set to indicate that the attachment appears inline within an item.</td>
</tr>
</tbody>
</table>

The **AttachmentType** complex type is extended by the **ItemAttachmentType** complex type, as specified in section 2.2.4.6, and the **FileAttachmentType** complex type, as specified in section 2.2.4.5.

### 2.2.4.5 t:FileAttachmentType Complex Type

The **FileAttachmentType** complex type represents a file that is attached to an item in the server store. The **FileAttachmentType** complex type extends the **AttachmentType** complex type, as specified in section 2.2.4.4.

```
<xs:complexType name="FileAttachmentType">
  <xs:complexContent>
    <xs:extension name="FileAttachmentType" base="t:AttachmentType">
      <xs:sequence>
        <xs:element name="IsContactPhoto" type="xs:boolean" minOccurs="0" maxOccurs="1" />
        <xs:element name="Content" type="xs:base64Binary" minOccurs="0" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
The following table lists the child elements of the `FileAttachmentType` complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsContactPhoto</td>
<td>xs:boolean</td>
<td>A value that indicates whether this attachment is a contact photo. A text value of &quot;true&quot; indicates that the attachment is a contact photo.</td>
</tr>
<tr>
<td>Content</td>
<td>xs:base64Binary</td>
<td>Represents the base64-encoded contents of the file attachment.</td>
</tr>
</tbody>
</table>

### 2.2.4.6 `t:ItemAttachmentType` Complex Type

The `ItemAttachmentType` complex type represents an item that is attached to another item in the server store. The `ItemAttachmentType` complex type extends the `AttachmentType` complex type as specified in section 2.2.4.4. If the item attachment is `MeetingMessage`, `MeetingRequest`, `MeetingResponse` or `MeetingCancellation`, the server SHOULD return `ErrorInvalidItemForOperationCreateItemAttachment` response code as specified in [MS-OXWSDATA] section 2.2.5.24. If none of the child elements of `ItemAttachmentType` is specified in the `CreateAttachment` request, the server MUST return an `ErrorMissingItemForCreateItemAttachment` response code as specified in [MS-OXWSDATA] section 2.2.5.24.

```xml
<xs:complexType name="ItemAttachmentType">
  <xs:complexContent>
    <xs:extension base="t:AttachmentType">
      <xs:choice minOccurs="0" maxOccurs="1">
        <xs:element name="Item" type="t:ItemType" />
        <xs:element name="Message" type="t:MessageType" />
        <xs:element name="CalendarItem" type="t:CalendarItemType" />
        <xs:element name="Contact" type="t:ContactItemType" />
        <xs:element name="MeetingMessage" type="t:MeetingMessageType" />
        <xs:element name="MeetingRequest" type="t:MeetingRequestMessageType" />
        <xs:element name="MeetingResponse" type="t:MeetingResponseMessageType" />
      </xs:choice>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
The following table lists the child elements of the ItemAttachmentType complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>t:ItemType ([MS-OXWSCORE] section 2.2.4.24)</td>
<td>Represents a generic item in the server store. All items of type t:ItemType SHOULD&lt;3&gt; be returned as Message of a t:MessageType type.</td>
</tr>
<tr>
<td>Message</td>
<td>t:MessageType ([MS-OXWSMSG] section 2.2.4.3)</td>
<td>Represents a server e-mail message.</td>
</tr>
<tr>
<td>CalendarItem</td>
<td>t:CalendarItemType ([MS-OXWSMTGS] section 2.2.4.6)</td>
<td>Represents a calendar item.</td>
</tr>
<tr>
<td>Contact</td>
<td>t:ContactItemType ([MS-OXWSCONT] section 2.2.4.3)</td>
<td>Represents a contact item.</td>
</tr>
<tr>
<td>MeetingMessage</td>
<td>t:MeetingMessageType ([MS-OXWSMTGS] section 2.2.4.15)</td>
<td>Represents a meeting message in the server store.</td>
</tr>
<tr>
<td>MeetingRequest</td>
<td>t:MeetingRequestMessageType ([MS-OXWSMTGS] section 2.2.4.17)</td>
<td>Represents a meeting request in the server store.</td>
</tr>
<tr>
<td>MeetingResponse</td>
<td>t:MeetingResponseMessageType ([MS-OXWSMTGS] section 2.2.4.18)</td>
<td>Represents a meeting response in the server store.</td>
</tr>
<tr>
<td>MeetingCancellation</td>
<td>t:MeetingCancellationMessageType ([MS-OXWSMTGS] section 2.2.4.14)</td>
<td>Represents a meeting cancellation in the server store.</td>
</tr>
<tr>
<td>Task</td>
<td>t:TaskType ([MS-OXWSTASK] section 2.2.4.6)</td>
<td>Represents a task in the server store.</td>
</tr>
<tr>
<td>PostItem</td>
<td>t:PostItemType ([MS-OXWSPOST] section 2.2.4.1)</td>
<td>Represents a post item in the server store.</td>
</tr>
</tbody>
</table>
Element | Type | Description
--- | --- | ---
RoleMember | t:RoleMemberItemType ([MS-OXWScore] section 2.2.4.43) | For internal use only.<4>
Network | t:NetworkItemType ([MS-OXWScore] section 2.2.4.30) | For internal use only.<5>
Person | t:AbchPersonItemType ([MS-OXWSCONT] section 2.2.4.1) | Represents a person in the server store.<6>
Booking | t:BookingItemType ([MS-OXWSDATA] section 2.2.4.20) | For internal use only.<7>

2.2.4.7 t:NonEmptyArrayOfRequestAttachmentIdsType Complex Type

The **NonEmptyArrayOfRequestAttachmentIdsType** complex type represents an array that contains attachment identifiers.

```xml
<xs:complexType name="NonEmptyArrayOfRequestAttachmentIdsType">
  <xs:choice minOccurs="1" maxOccurs="unbounded">
    <xs:element name="AttachmentId" type="t:RequestAttachmentIdType" />
  </xs:choice>
</xs:complexType>
```

The following table lists the child element of the **NonEmptyArrayOfRequestAttachmentIdsType** complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentId</td>
<td>t:RequestAttachmentIdType (section 2.2.4.9)</td>
<td>Represents an identifier for an attachment.</td>
</tr>
</tbody>
</table>

2.2.4.8 t:ReferenceAttachmentType Complex Type

The **ReferenceAttachmentType** complex type represents a reference attachment. This type extends the **AttachmentType** complex type (section 2.2.4). <8>

```xml
<xs:complexType name="ReferenceAttachmentType">
  <xs:complexContent>
    <xs:extension base="t:AttachmentType">
      <xs:sequence>
        <xs:element name="AttachLongPathName" type="xs:string" minOccurs="0" maxOccurs="1" />
        <xs:element name="ProviderType" type="xs:string" minOccurs="0" maxOccurs="1" />
        <xs:element name="ProviderEndpointUrl" type="xs:string" minOccurs="0" maxOccurs="1" />
        <xs:element name="AttachmentThumbnailUrl" type="xs:string" minOccurs="0" maxOccurs="1" />
        <xs:element name="AttachmentPreviewUrl" type="xs:string" minOccurs="0" maxOccurs="1" />
        <xs:element name="PermissionType" type="xs:int" minOccurs="0" maxOccurs="1" />
        <xs:element name="AttachmentIsFolder" type="xs:boolean" minOccurs="0" maxOccurs="1" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
The following table lists the child elements of the `ReferenceAttachmentType` complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachLongPathName</td>
<td><code>xs:string</code> ([XMLSCHEMA2])</td>
<td>Specifies the URL of the attachment.</td>
</tr>
<tr>
<td>ProviderType</td>
<td><code>xs:string</code></td>
<td>Specifies the provider type.</td>
</tr>
<tr>
<td>ProviderEndpointUrl</td>
<td><code>xs:string</code></td>
<td>Specifies the URL of the provider endpoint.</td>
</tr>
<tr>
<td>AttachmentThumbnailUrl</td>
<td><code>xs:string</code></td>
<td>Specifies the URL of the thumbnail of the attachment.</td>
</tr>
<tr>
<td>AttachmentPreviewUrl</td>
<td><code>xs:string</code></td>
<td>Specifies the URL of the attachment preview.</td>
</tr>
<tr>
<td>PermissionType</td>
<td><code>xs:int</code> ([XMLSCHEMA2])</td>
<td>Specifies the permission type.</td>
</tr>
<tr>
<td>AttachmentIsFolder</td>
<td><code>xs:Boolean</code> ([XMLSCHEMA2])</td>
<td>Specifies that the attachment is a folder.</td>
</tr>
</tbody>
</table>

2.2.4.9 `t:RequestAttachmentIdType` Complex Type

The `RequestAttachmentIdType` complex type represents an identifier for an attachment. The `RequestAttachmentIdType` complex type extends the `BaseItemIdType` complex type ([MS-OXWSCTDA] section 2.2.4.15).

```xml
<xs:complexType name="RequestAttachmentIdType">
  <xs:complexContent>
    <xs:extension base="t:BaseItemIdType">
      <xs:attribute name="Id" type="xs:string" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table summarizes the set of common XML schema attribute definitions defined by this specification.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td><code>xs:string</code> ([XMLSCHEMA2])</td>
<td>Provides an identifier for an attachment. The maximum length is 512 bytes after base64 decoding.</td>
</tr>
</tbody>
</table>
2.2.5 Simple Types
This specification does not define any common XML schema simple type definitions.

2.2.6 Attributes
This specification does not define any common XML schema attribute definitions.

2.2.7 Groups
This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups
This specification does not define any common XML schema attribute group definitions.
3 Protocol Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.1 ExchangeServicePortType Server Details

The Attachment Handling Web Service Protocol defines a single WSDL port type with three operations. The operations enable client implementations to create, delete, and get attachments.

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 specified in this document.

The Attachment Handling Web Service Protocol is used to create, delete, and get attachments on items on the account’s mailbox on the server.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of WSDL operations as defined by this specification:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateAttachment</td>
<td>Creates either an item or file attachment and attaches it to the specified item.</td>
</tr>
<tr>
<td>DeleteAttachment</td>
<td>Deletes file and item attachments from an existing item in the server store.</td>
</tr>
<tr>
<td>GetAttachment</td>
<td>Retrieves existing attachments on items in the server store.</td>
</tr>
</tbody>
</table>

3.1.4.1 CreateAttachment Operation

The CreateAttachment operation creates an item or file attachment on an item in the server store.

The following is the WSDL port type specification of the CreateAttachment operation.

```xml
<wsdl:operation name="CreateAttachment">
  <wsdl:input message="tns:CreateAttachmentSoapIn" />
  <wsdl:output message="tns:CreateAttachmentSoapOut" />
</wsdl:operation>
```
The following is the WSDL binding specification of the CreateAttachment operation.

```xml
<wsdl:operation name="CreateAttachment">
  <wsdl:input>
    <soap:header message="tns:CreateAttachmentSoapIn" part="Impersonation" use="literal"/>
    <soap:header message="tns:CreateAttachmentSoapIn" part="MailboxCulture" use="literal"/>
    <soap:header message="tns:CreateAttachmentSoapIn" part="RequestVersion" use="literal"/>
    <soap:header message="tns:CreateAttachmentSoapIn" part="TimeZoneContext" use="literal"/>
    <soap:body parts="request" use="literal" />
  </wsdl:input>
  <wsdl:output>
    <soap:body parts="CreateAttachmentResult" use="literal" />
    <soap:header message="tns:CreateAttachmentSoapOut" part="ServerVersion" use="literal"/>
  </wsdl:output>
</wsdl:operation>
```

The protocol client sends a CreateAttachmentSoapIn request WSDL message, and the protocol server responds with a CreateAttachmentSoapOut response WSDL message.

An item attachment does not exist as a store item. It only exists as an attachment on an item or another attachment. Item attachments can be retrieved only by using the GetAttachment operation request.

### 3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to the CreateAttachment operation.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateAttachmentSoapIn</td>
<td>Specifies the SOAP message that creates an attachment.</td>
</tr>
<tr>
<td>CreateAttachmentSoapOut</td>
<td>Specifies the SOAP message that is returned by the server in response.</td>
</tr>
</tbody>
</table>

#### 3.1.4.1.1.1 tns:CreateAttachmentSoapIn Message

The CreateAttachmentSoapIn WSDL message specifies the CreateAttachment operation request to create an attachment.

```xml
<wsdl:message name="CreateAttachmentSoapIn">
  <wsdl:part name="request" element="tns:CreateAttachment" />
  <wsdl:part name="Impersonation" element="t:ExchangeImpersonation"/>
  <wsdl:part name="MailboxCulture" element="t:MailboxCulture"/>
  <wsdl:part name="RequestVersion" element="t:RequestServerVersion"/>
  <wsdl:part name="TimeZoneContext" element="t:TimeZoneContext"/>
</wsdl:message>
```

The CreateAttachmentSoapIn WSDL message is the input message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/CreateAttachment.

The five parts of the CreateAttachmentSoapIn WSDL message are described in the following table.
<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>tns:CreateAttachment (section 3.1.4.1.2.1)</td>
<td>Specifies the SOAP body of the request create an attachment.</td>
</tr>
<tr>
<td>Impersonation</td>
<td>t:ExchangeImpersonation ([MS-OXWSCDATA] section 2.2.3.3)</td>
<td>Specifies a SOAP header that identifies the user whom the client application is impersonating.</td>
</tr>
<tr>
<td>MailboxCulture</td>
<td>t:MailboxCulture ([MS-OXWSCDATA] section 2.2.3.6)</td>
<td>Specifies a SOAP header that identifies the culture to use for accessing the mailbox. The cultures are defined by [RFC3066].</td>
</tr>
<tr>
<td>RequestVersion</td>
<td>t:RequestServerVersion ([MS-OXWSCDATA] section 2.2.3.9)</td>
<td>Specifies a SOAP header that identifies the schema version for the CreateAttachment operation request.</td>
</tr>
<tr>
<td>TimeZoneContext</td>
<td>t:TimeZoneContext ([MS-OXWSGTZ] section 2.2.3.4)</td>
<td>Specifies a time zone definition and enables associating SOAP attributes with the definition.</td>
</tr>
</tbody>
</table>

### 3.1.4.1.2 tns:CreateAttachmentSoapOut Message

The CreateAttachmentSoapOut WSDL message specifies the server response to the CreateAttachment operation request to create an attachment.

```xml
<wsdl:message name="CreateAttachmentSoapOut">
  <wsdl:part name="CreateAttachmentResult" element="tns:CreateAttachmentResponse" />
  <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
</wsdl:message>
```

The CreateAttachmentSoapOut WSDL message is the output message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/CreateAttachment.

The two parts of the CreateAttachmentSoapOut WSDL message are described in the following table.

<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateAttachmentResult</td>
<td>tns:CreateAttachmentResponse (section 3.1.4.1.2.2)</td>
<td>Specifies the SOAP body of the response to a CreateAttachment operation request.</td>
</tr>
<tr>
<td>ServerVersion</td>
<td>t:ServerVersionInfo ([MS-OXWSCDATA] section 2.2.3.10)</td>
<td>Specifies a SOAP header that identifies the server version for the response.</td>
</tr>
</tbody>
</table>

If the request is successful, the CreateAttachment operation returns a CreateAttachmentResponse element with the ResponseClass attribute of the CreateAttachmentResponseMessage element set to "Success". The ResponseCode element of the CreateAttachmentResponse element is set to "NoError".

If the request is unsuccessful, the CreateAttachment operation returns a CreateAttachmentResponse element with the ResponseClass attribute of the CreateAttachmentResponseMessage element set to "Error". The ResponseCode element of the CreateAttachmentResponseMessage element is set to a value of the ResponseCodeType simple type, as specified in [MS-OXWSCDATA] section 2.2.5.24.

### 3.1.4.1.2 Elements
The following table summarizes the XML schema element definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateAttachment</td>
<td>Specifies a request to create an attachment.</td>
</tr>
<tr>
<td>CreateAttachmentResponse</td>
<td>Specifies the response body content from a request to create an attachment.</td>
</tr>
</tbody>
</table>

### 3.1.4.1.2.1 CreateAttachment Element

The **CreateAttachment** element specifies the request message for a CreateAttachment operation.

```xml
<xs:element name="CreateAttachment"
    type="m:CreateAttachmentType"/>
```

### 3.1.4.1.2.2 CreateAttachmentResponse Element

The **CreateAttachmentResponse** element specifies the response message for a CreateAttachment operation (section 3.1.4.1).

```xml
<xs:element name="CreateAttachmentResponse"
    type="m:CreateAttachmentResponseType"/>
```

### 3.1.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateAttachmentResponseType</td>
<td>Specifies a response message for the CreateAttachment operation.</td>
</tr>
<tr>
<td>CreateAttachmentType</td>
<td>Specifies request message for the CreateAttachment operation.</td>
</tr>
</tbody>
</table>

#### 3.1.4.1.3.1 m:CreateAttachmentResponseType Complex Type

The **CreateAttachmentResponseType** complex type specifies the response message that is returned by the CreateAttachment operation. The **CreateAttachmentResponseType** complex type extends the **BaseResponseMessageType** complex type ([MS-OXWSCDATA] section 2.2.4.18).

```xml
<xs:complexType name="CreateAttachmentResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
```
3.1.4.1.3.2  m:CreateAttachmentType Complex Type

The CreateAttachmentType complex type specifies a request message to attach an item or file to a specified item in the server database. The CreateAttachmentType complex type extends the BaseRequestType complex type ([MS-OXWSCDATA] section 2.2.4.17).

```xml
<xs:complexType name="CreateAttachmentType">
  <xs:complexContent>
    <xs:extension base="m:BaseRequestType">
      <xs:sequence>
        <xs:element name="ParentItemId" type="t:ItemIdType"/>
        <xs:element name="Attachments" type="t:NonEmptyArrayOfAttachmentsType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table lists the child elements of the CreateAttachmentType complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ParentItemId</td>
<td>t:ItemIdType ([MS-OXWSCORE] section 2.2.4.25)</td>
<td>Identifies the parent item in the server store that contains the attachment. The ParentItemId element MUST provide the ID of a store item. The ParentItemId element can be retrieved by using the GetItem operation ([MS-OXWSCORE] section 3.1.4.4).</td>
</tr>
<tr>
<td>Attachments</td>
<td>t:NonEmptyArrayOfAttachmentsType ([MS-OXWSCDATA] section 2.2.4.49)</td>
<td>Contains the items or files that are attached to an item in the server store.</td>
</tr>
</tbody>
</table>

3.1.4.2  DeleteAttachment Operation

The DeleteAttachment operation deletes attachments from an item in the server store.

The following is the WSDL port type specification of the operation.

```xml
<wsdl:operation name="DeleteAttachment">
  <wsdl:input message="tns:DeleteAttachmentSoapIn" />
  <wsdl:output message="tns:DeleteAttachmentSoapOut" />
</wsdl:operation>
```

The following is the WSDL binding specification of the operation.

```xml
<wsdl:operation name="DeleteAttachment">
  <wsdl:input>
    <soap:header message="tns:DeleteAttachmentSoapIn" part="Impersonation" use="literal"/>
  </wsdl:input>
```
The protocol client sends a `DeleteAttachmentSoapIn` request WSDL message, and the protocol server responds with a `DeleteAttachmentSoapOut` response WSDL message.

An item attachment does not exist as a store item. It only exists as an attachment on an item or another attachment. Item attachments can be retrieved only by using the `GetAttachment` operation request. Before an item attachment can be deleted, it MUST be retrieved from the server.

### 3.1.4.2.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DeleteAttachmentSoapIn</code></td>
<td>Specifies the SOAP message that deletes an attachment.</td>
</tr>
<tr>
<td><code>DeleteAttachmentSoapOut</code></td>
<td>Specifies the SOAP message that is returned by the server in response.</td>
</tr>
</tbody>
</table>

#### 3.1.4.2.1.1 `tns:DeleteAttachmentSoapIn` Message

The `DeleteAttachmentSoapIn` WSDL message specifies the `DeleteAttachment` operation request to delete an attachment.

```xml
<wsdl:message name="DeleteAttachmentSoapIn">
  <wsdl:part name="request" element="tns:DeleteAttachment"/>
  <wsdl:part name="Impersonation" element="t:ExchangeImpersonation"/>
  <wsdl:part name="MailboxCulture" element="t:MailboxCulture"/>
  <wsdl:part name="RequestVersion" element="t:RequestServerVersion"/>
</wsdl:message>
```

The `DeleteAttachmentSoapIn` WSDL message is the input message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/DeleteAttachment.

The four parts of the `DeleteAttachmentSoapIn` WSDL message are described in the following table.

<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td><code>tns:DeleteAttachment</code> (section 3.1.4.2.2.1)</td>
<td>Specifies the SOAP body of the request to delete an attachment.</td>
</tr>
<tr>
<td>Impersonation</td>
<td><code>t:ExchangeImpersonation</code> ([MS-OXWSCDATA] section 2.2.3.3)</td>
<td>Specifies a SOAP header that identifies the user who the client application is impersonating.</td>
</tr>
<tr>
<td>MailboxCulture</td>
<td><code>t:MailboxCulture</code> ([MS-OXWSCDATA])</td>
<td>Specifies a SOAP header that identifies the culture</td>
</tr>
</tbody>
</table>
### 3.1.4.2.1.2  tns:DeleteAttachmentSoapOut Message

The **DeleteAttachmentSoapOut** WSDL message specifies the server response to the **DeleteAttachment** operation request to delete an attachment.

```xml
<wSDL:message name="DeleteAttachmentSoapOut">
  <wSDL:part name="DeleteAttachmentResult" element="tns:DeleteAttachmentResponse" />
  <wSDL:part name="ServerVersion" element="t:ServerVersionInfo"/>
</wSDL:message>
```

The **DeleteAttachmentSoapOut** WSDL message is the output message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/DeleteAttachment.

The two parts of the **DeleteAttachmentSoapOut** WSDL message are described in the following table.

<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteAttachmentResult</td>
<td>tns:DeleteAttachmentResponse</td>
<td>Specifies the SOAP body of the response to a <strong>DeleteAttachment</strong> operation request.</td>
</tr>
<tr>
<td></td>
<td>(section 3.1.4.2.2)</td>
<td></td>
</tr>
<tr>
<td>ServerVersion</td>
<td>t:ServerVersionInfo ([MS-OXWSCDATA] section 2.2.3.10)</td>
<td>Specifies a SOAP header that identifies the server version for the response.</td>
</tr>
</tbody>
</table>

If the request is successful, the **DeleteAttachment** operation returns a **DeleteAttachmentResponse** element with the **ResponseClass** attribute of the **DeleteAttachmentResponseMessage** element set to "Success". The **ResponseCode** element of the **DeleteAttachmentResponse** element is set to "No Error".

If the request is unsuccessful, the **DeleteAttachment** operation returns a **DeleteAttachmentResponse** element with the **ResponseClass** attribute of the **DeleteAttachmentResponseMessage** element set to "Error". The **ResponseCode** element of the **DeleteAttachmentResponseMessage** element is set to a value of the **ResponseCodeType** simple type, as specified in [MS-OXWSCDATA] section 2.2.5.24.

### 3.1.4.2.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteAttachment</td>
<td>Specifies a request to delete an attachment.</td>
</tr>
<tr>
<td>DeleteAttachmentResponse</td>
<td>Specifies the response body content from a request to delete an attachment.</td>
</tr>
</tbody>
</table>
3.1.4.2.2.1 DeleteAttachment Element

The DeleteAttachment element specifies the request message for a DeleteAttachment operation.

```xml
<xs:element name="DeleteAttachment" type="m:DeleteAttachmentType" />
```

3.1.4.2.2.2 DeleteAttachmentResponse Element

The DeleteAttachmentResponse element specifies the response message for a DeleteAttachment operation.

```xml
<xs:element name="DeleteAttachmentResponse" type="m:DeleteAttachmentResponseType" />
```

3.1.4.2.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteAttachmentResponseMessageType</td>
<td>Contains the status and result of a single DeleteAttachment operation.</td>
</tr>
<tr>
<td>DeleteAttachmentResponseType</td>
<td>Specifies a response message for the DeleteAttachment operation.</td>
</tr>
<tr>
<td>DeleteAttachmentType</td>
<td>Specifies a request message for the DeleteAttachment operation.</td>
</tr>
<tr>
<td>RootItemIdType</td>
<td>Identifies the root item of a deleted attachment. Because an attachment can be attached to another attachment, the root item is the store item that owns the entire attachment hierarchy.</td>
</tr>
</tbody>
</table>

3.1.4.2.3.1 m:DeleteAttachmentResponseMessageType Complex Type

The DeleteAttachmentResponseMessageType complex type contains the status and result of a single DeleteAttachment operation. The DeleteAttachmentResponseMessageType complex type extends the ResponseMessageType complex type ([MS-OXWSCDATA] section 2.2.4.67).

```xml
<xs:complexType name="DeleteAttachmentResponseMessageType">
  <xs:complexContent>
    <xs:extension base="m:ResponseMessageType">
      <xs:sequence>
        <xs:element name="RootItemId" type="t:RootItemIdType" minOccurs="0" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
The following table lists the child element of the `DeleteAttachmentResponseMessageType` complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RootItemId</td>
<td><code>t:RootItemIdType</code> (section 3.1.4.2.3.4)</td>
<td>Specifies the parent item of a deleted attachment.</td>
</tr>
</tbody>
</table>

**3.1.4.2.3.2 m:DeleteAttachmentResponseType Complex Type**

The `DeleteAttachmentResponseType` complex type specifies the response message that is returned by the `DeleteAttachment` operation. The `DeleteAttachmentResponseType` complex type extends the `BaseResponseMessageType` complex type ([MS-OXWSCDATA] section 2.2.4.18).

```xml
<x:simpleType name="DeleteAttachmentResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
```

**3.1.4.2.3.3 m:DeleteAttachmentType Complex Type**

The `DeleteAttachmentType` complex type specifies a request message to delete attachments on an item in the server database. The `DeleteAttachmentType` complex type extends the `BaseRequestType` complex type ([MS-OXWSCDATA] section 2.2.4.17).

```xml
<x:simpleType name="DeleteAttachmentType">
    <xs:complexContent>
        <xs:extension base="m:BaseRequestType">
            <xs:sequence>
                <xs:element name="AttachmentIds" type="t:NonEmptyArrayOfRequestAttachmentIdsType"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The following table lists the child element of the `DeleteAttachmentType` complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentIds</td>
<td><code>t:NonEmptyArrayOfRequestAttachmentIdsType</code> (section 2.2.4.7)</td>
<td>Contains the identifiers of the items or files that are attached to an item in the server store to be deleted.</td>
</tr>
</tbody>
</table>
### 3.1.4.2.3.4 t:RootItemIdType Complex Type

The `RootItemIdType` complex type identifies the root item of a deleted attachment and the new change key to the parent item. The `RootItemIdType` complex type extends the `BaseItemIdType` complex type ([MS-OXWSCDATA] section 2.2.4.15).

```xml
<xs:complexType name="RootItemIdType">
  <xs:complexContent>
    <xs:extension base="t:BaseItemIdType">
      <xs:attribute name="RootItemId" type="xs:string" use="required" />
      <xs:attribute name="RootItemChangeKey" type="xs:string" use="required" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The following table summarizes the set of common XML schema attribute definitions defined by this specification.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RootItemId</td>
<td>xs:string</td>
<td>Identifies the root item of an attachment. The maximum length is 512 bytes after base64 decoding.</td>
</tr>
<tr>
<td>RootItemChangeKey</td>
<td>xs:string</td>
<td>Identifies the new change key of the root item of an attachment. The maximum length is 512 bytes after base64 decoding.</td>
</tr>
</tbody>
</table>

### 3.1.4.3 GetAttachment Operation

The `GetAttachment` operation gets attachments from an item in the server store.

The following is the WSDL port type specification of the operation.

```xml
<wsdl:operation name="GetAttachment">
  <wsdl:input message="tns:GetAttachmentSoapIn" />
  <wsdl:output message="tns:GetAttachmentSoapOut" />
</wsdl:operation>
```

The following is the WSDL binding specification of the operation.

```xml
<wsdl:operation name="GetAttachment">
```
The protocol client sends a GetAttachmentSoapIn request WSDL message, and the protocol server responds with a GetAttachmentSoapOut response WSDL message.

### 3.1.4.3.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAttachmentSoapIn</td>
<td>Specifies the SOAP message that gets an attachment.</td>
</tr>
<tr>
<td>GetAttachmentSoapOut</td>
<td>Specifies the SOAP message that is returned by the server in response.</td>
</tr>
</tbody>
</table>

#### 3.1.4.3.1.1 tns:GetAttachmentSoapIn Message

The GetAttachmentSoapIn WSDL message specifies the GetAttachment operation request to get an attachment.

```xml
<wsdl:message name="GetAttachmentSoapIn">
  <wsdl:part name="request" element="tns:GetAttachment" />
  <wsdl:part name="Impersonation" element="t:ExchangeImpersonation"/>
  <wsdl:part name="MailboxCulture" element="t:MailboxCulture"/>
  <wsdl:part name="RequestVersion" element="t:RequestServerVersion"/>
  <wsdl:part name="TimeZoneContext" element="t:TimeZoneContext"/>
</wsdl:message>
```

The GetAttachmentSoapIn WSDL message is the input message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/GetAttachment.

The five parts of the GetAttachmentSoapIn message are described in the following table.

<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>tns:GetAttachment (section 3.1.4.3.2.1)</td>
<td>Specifies the SOAP body of the request to get an attachment.</td>
</tr>
<tr>
<td>Impersonation</td>
<td>t:ExchangeImpersonation ([MS-OXWSDATA] section 2.2.3.3)</td>
<td>Specifies a SOAP header that identifies the user who the client application is impersonating.</td>
</tr>
<tr>
<td>MailboxCulture</td>
<td>t:MailboxCulture ([MS-OXWSDATA] section 2.2.3.6)</td>
<td>Specifies a SOAP header that identifies the culture to use for accessing the mailbox.</td>
</tr>
</tbody>
</table>
### 3.1.4.3.1.2  

tns:GetAttachmentSoapOut Message

The GetAttachmentSoapOut WSDL message specifies the server response to the GetAttachment operation request to get an attachment.

```xml
<wSDL:message name="GetAttachmentSoapOut">
  <wSDL:part name="GetAttachmentResult" element="tns:GetAttachmentResponse" />
  <wSDL:part name="ServerVersion" element="t:ServerVersionInfo"/>
</wSDL:message>
```

The GetAttachmentSoapOut WSDL message is the output message for the SOAP action http://schemas.microsoft.com/exchange/services/2006/messages/GetAttachment.

The two parts of the GetAttachmentSoapOut WSDL message are described in the following table.

<table>
<thead>
<tr>
<th>Part</th>
<th>Element/type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAttachmentResult</td>
<td>tns:GetAttachmentResponse (section 3.1.4.3.2.2)</td>
<td>Specifies the SOAP body of the response to a GetAttachment operation request.</td>
</tr>
<tr>
<td>ServerVersion</td>
<td>t:ServerVersionInfo ([MS-OXWSCDATA] section 2.2.3.10)</td>
<td>Specifies a SOAP header that identifies the server version for the response.</td>
</tr>
</tbody>
</table>

If the request is successful, the GetAttachment operation returns a GetAttachmentResponse element with the ResponseClass attribute of the GetAttachmentResponseMessage element set to "Success". The ResponseCode element of the GetAttachmentResponse element is set to "NoError".

If the request is unsuccessful, the GetAttachment operation returns a GetAttachmentResponse element with the ResponseClass attribute of the GetAttachmentResponseMessage element set to "Error". The ResponseCode element of the GetAttachmentResponseMessage element is set to a value of the ResponseCodeType simple type, as specified in [MS-OXWSCDATA] section 2.2.5.24.

### 3.1.4.3.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAttachment</td>
<td>Specifies a request to get an attachment.</td>
</tr>
<tr>
<td>GetAttachmentResponse</td>
<td>Specifies the response body content from a request to get an attachment.</td>
</tr>
</tbody>
</table>
3.1.4.3.2.1 GetAttachment Element

The GetAttachment element specifies the request message for a GetAttachment operation.

```xml
<xs:element name="GetAttachment"
    type="m:GetAttachmentType"/>
```

3.1.4.3.2.2 GetAttachmentResponse Element

The GetAttachmentResponse element specifies the response message to a GetAttachment operation.

```xml
<xs:element name="GetAttachmentResponse"
    type="m:GetAttachmentResponseType"/>
```

3.1.4.3.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAttachmentResponseType</td>
<td>Specifies a response message for the GetAttachment operation.</td>
</tr>
<tr>
<td>GetAttachmentType</td>
<td>Specifies a request message for the GetAttachment operation.</td>
</tr>
<tr>
<td>AttachmentResponseShapeType</td>
<td>Specifies additional properties for the GetAttachment operation to return.</td>
</tr>
</tbody>
</table>

3.1.4.3.3.1 m:GetAttachmentResponseType Complex Type

The GetAttachmentResponseType complex type specifies the response message that is returned by the GetAttachment operation. The GetAttachmentResponseType complex type extends the BaseResponseMessageType complex type ([MS-OXWSCDATA] section 2.2.4.18).

```xml
<xs:complexType name="GetAttachmentResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
```

3.1.4.3.3.2 m:GetAttachmentType Complex Type

The GetAttachmentType complex type specifies a request message to get attached items and files on an item in the server database. The GetAttachmentType complex type extends the BaseRequestType complex type ([MS-OXWSCDATA] section 2.2.4.17).

```xml
<xs:complexType name="GetAttachmentType">
    <xs:complexContent>
</xs:complexType>
```
The following table lists the child elements of the **GetAttachmentType** complex type.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentShape</td>
<td>t:AttachmentResponseShapeType (section 3.1.4.3.3.3)</td>
<td>Contains additional properties to return for the attachments.</td>
</tr>
<tr>
<td>AttachmentIds</td>
<td>t:NonEmptyArrayOfRequestAttachmentIdsType (section 2.2.4.7)</td>
<td>Contains the identifiers of the attachments to return in the response.</td>
</tr>
</tbody>
</table>

### 3.1.4.3.3.3  t:AttachmentResponseShapeType Complex Type

The **AttachmentResponseShapeType** complex type identifies additional properties for the **GetAttachment** operation to return.

```xml
<xs:complexType name="AttachmentResponseShapeType">
  <xs:sequence>
    <xs:element name="IncludeMimeContent" type="xs:boolean" minOccurs="0" />
    <xs:element name="BodyType" type="t:BodyTypeResponseType" minOccurs="0" />
    <xs:element name="FilterHtmlContent" type="xs:boolean" minOccurs="0" maxOccurs="1" />
    <xs:element name="AdditionalProperties" type="t:NonEmptyArrayOfPathsToElementType" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

The following table lists the child elements of the **AttachmentResponseShapeType** complex type. <9>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncludeMimeContent</td>
<td>xs:boolean ([XMLSCHEMA2])</td>
<td>Indicates whether the MIME content of an item or attachment is returned in a response. A text value of &quot;true&quot; indicates that the attachment contains MIME content.</td>
</tr>
<tr>
<td>BodyType</td>
<td>t:BodyTypeResponseType ([MS-OXWSCDATA] section 2.2.5.1)</td>
<td>Represents the format of the body text in a response.</td>
</tr>
<tr>
<td>FilterHtmlContent</td>
<td>xs:boolean ([XMLSCHEMA2])</td>
<td>Indicates whether to filter potentially unsafe HTML content from message bodies. A text value of &quot;true&quot; indicates that potentially unsafe HTML content is to be filtered from the attachment.</td>
</tr>
<tr>
<td>AdditionalProperties</td>
<td>t:NonEmptyArrayOfPathsToElementType ([MS-OXWSCDATA] section 2.2.4.50)</td>
<td>Contains additional properties to return in a response.</td>
</tr>
</tbody>
</table>

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

3.2 Client Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Message Processing Events and Sequencing Rules

None.
3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.
4 Protocol Examples

The following examples show the request and response XML for the Attachment Handling Web Service Protocol operations.

4.1 CreateAttachment Example

The following is an example of a CreateAttachment operation, which creates an attachment named Deleteme.txt on the specified message. The item that it is attached to, which is represented by the ParentItemId element, is created separately. The Id attribute of the ParentItemId element has been shortened to preserve readability.

The client constructs the request XML and sends it to the server.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
   xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <t:RequestServerVersion Version="Exchange2010" />
  </soap:Header>
  <soap:Body>
    <m:CreateAttachment>
      <m:ParentItemId Id="AAMkAGIwODEy" />
      <m:Attachments>
        <t:FileAttachment>
          <t:Name>Deleteme.txt</t:Name>
          <t:IsInline>false</t:IsInline>
          <t:IsContactPhoto>false</t:IsContactPhoto>
          <t:Content>UGxlYXNlIGRlbGV0ZSB0aGlzIGZpbGUu</t:Content>
        </t:FileAttachment>
      </m:Attachments>
    </m:CreateAttachment>
  </soap:Body>
</soap:Envelope>
```

The server constructs the response XML and sends it to the client. The Id, RootItemId, and RootItemChangeKey attributes of the AttachmentId element have been shortened to preserve readability.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <h:ServerVersionInfo MajorVersion="14"
      MinorVersion="1" MajorBuildNumber="63"
      MinorBuildNumber="0"
      Version="Exchange2010"
      xmlns:h="http://schemas.microsoft.com/exchange/services/2006/types"
      xmlns="http://schemas.microsoft.com/exchange/services/2006/types"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"/>
  </s:Header>
  <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <m:CreateAttachmentResponse
      xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
      xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
      <m:ResponseMessages>
        <m:CreateAttachmentResponseMessage ResponseClass="Success">
          <m:ResponseCode>NoError</m:ResponseCode>
          <m:Attachments>
            <t:FileAttachment>
```

[MS-OXWSATT] - v20220215
Attachment Handling Web Service Protocol
Copyright © 2022 Microsoft Corporation
Release: February 15, 2022
4.2 DeleteAttachment Example

The following is an example of a **DeleteAttachment** operation. This example deletes an attachment from an item.

The client has to have the identifier for the item that has the attachment. The client constructs the request **XML** and sends it to the server. The **Id** attribute of the **AttachmentId** element has been shortened to preserve readability.

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
    xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Header>
        <t:RequestServerVersion Version="Exchange2010" />
    </soap:Header>
    <soap:Body>
        <m:DeleteAttachment>
            <m:AttachmentIds>
                <t:AttachmentId Id="AAMkAGY4YzQw" />
            </m:AttachmentIds>
        </m:DeleteAttachment>
    </soap:Body>
</soap:Envelope>
```

The server constructs the response XML and sends it to the client. The **RootItemId** and **RootItemChangeKey** attributes of the **RootItemId** element have been shortened to preserve readability.

```xml
<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
    <s:Header>
        <s:ServerVersionInfo MajorVersion="14"
            MinorVersion="1"
            MajorBuildNumber="63" MinorBuildNumber="0"
            Version="Exchange2010"
            xmlns:h="http://schemas.microsoft.com/exchange/services/2006/types"
            xmlns="http://schemas.microsoft.com/exchange/services/2006/types"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema" />
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <m:DeleteAttachmentResponse xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
            xmlns:xm="http://schemas.microsoft.com/exchange/services/2006/types">
            <m:ResponseMessages>
                <m:DeleteAttachmentResponseMessage xsi:type="m:DeleteAttachmentResponseMessageType" ResponseClass="Success">
                    <m:ResponseCode>NoError</m:ResponseCode>
                    <m:RootItemId RootItemId="AAMkAGY4YzQw" RootItemChangeKey="CQAAABYAAA" />
                </m:DeleteAttachmentResponseMessage>
            </m:ResponseMessages>
        </m:DeleteAttachmentResponse>
    </s:Body>
</s:Envelope>
```
4.3 GetAttachment Example

The following is an example of a GetAttachment operation. This example gets an attachment from the server store.

The client constructs the request XML and sends it to the server. The Id attribute of the AttachmentId element has been shortened to preserve readability.

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
    xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Header>
        <t:RequestServerVersion Version="Exchange2010" />
    </soap:Header>
    <soap:Body>
        <m:GetAttachment>
            <m:AttachmentIds>
                <t:AttachmentId Id="AAMkAGY4YzQw" />
            </m:AttachmentIds>
        </m:GetAttachment>
    </soap:Body>
</soap:Envelope>
```

The server constructs the response XML and sends it to the client. In this example, there is one attachment in the attachments collection. The Id attribute of the AttachmentId element has been shortened to preserve readability. Note that the content is base64.

```xml
<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
    <s:Header>
        <h:ServerVersionInfo MajorVersion="14"
            MinorVersion="1"
            MajorBuildNumber="63"
            MinorBuildNumber="0"
            Version="Exchange2010"
            xmlns:h="http://schemas.microsoft.com/exchange/services/2006/types"
            xmlns="http://schemas.microsoft.com/exchange/services/2006/types"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema" />
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <m:GetAttachmentResponse xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
            xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
            <m:ResponseMessages>
                <m:GetAttachmentResponseMessage ResponseClass="Success">
                    <m:ResponseCode>NoError</m:ResponseCode>
                    <m:Attachments>
                        <t:AttachmentId Id="AAMkAGY4YzQw" />
                        <t:Name>Deleteme.txt</t:Name>
                        <t:Content>UGxlYXNlIGRlbGV0ZSB0aGlzIHRleHQu</t:Content>
                    </m:Attachments>
                </m:GetAttachmentResponseMessage>
            </m:ResponseMessages>
        </m:GetAttachmentResponse>
    </s:Body>
</s:Envelope>
```
</m:ResponseMessages>
</m:GetAttachmentResponse>
</s:Body>
</s:Envelope>
5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.
6 Appendix A: Full WSDL

The following table lists the XML files that are required to implement the functionality that is specified in this document.

<table>
<thead>
<tr>
<th>File name</th>
<th>Description</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-OXWSATT.wsdl</td>
<td>Contains the WSDL for the implementation of this protocol.</td>
<td>6</td>
</tr>
<tr>
<td>MS-OXWSATT-messages.xsd</td>
<td>Contains the XML schema message definitions that are used in this protocol.</td>
<td>7.1</td>
</tr>
<tr>
<td>MS-OXWSATT-types.xsd</td>
<td>Contains the XML schema type definitions that are used in this protocol.</td>
<td>7.2</td>
</tr>
</tbody>
</table>

These files have to be placed in a common folder in order for the WSDL to validate and operate. Also, any schema files that are included in or imported into the MS-OXWSATT-types.xsd schema or the MS-OXWSATT-messages.xsd schema have to be placed in the common folder with these files.

This section contains the contents of the MS-OXWSATT.wsdl file.

```xml
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/
xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/messages"
xmlns:targetNamespace="http://schemas.microsoft.com/exchange/services/2006/messages">
<wsdl:types>
  <xs:schema id="messages" elementFormDefault="qualified" version="Exchange2016"
xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/messages"
xmlns:xsi="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.microsoft.com/exchange/services/2006/messages">
    <xs:include schemaLocation="MS-OXWSATT-messages.xsd" />
  </xs:schema>
  <xs:schema id="types" elementFormDefault="qualified" version="Exchange2016"
xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/types"
targetNamespace="http://schemas.microsoft.com/exchange/services/2006/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema"
xmlns:targetNamespace="http://schemas.microsoft.com/exchange/services/2006/messages">
    <xs:include schemaLocation="MS-OXWSATT-types.xsd" />
  </xs:schema>
</wsdl:types>
<wsdl:message name="CreateAttachmentSoapIn" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:part name="request" element="tns:CreateAttachment"/>
  <wsdl:part name="Impersonation" element="t:ExchangeImpersonation"/>
  <wsdl:part name="MailboxCulture" element="t:MailboxCulture"/>
  <wsdl:part name="RequestVersion" element="t:RequestServerVersion"/>
  <wsdl:part name="TimeZoneContext" element="t:TimeZoneContext"/>
</wsdl:message>
<wsdl:message name="CreateAttachmentSoapOut" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:part name="CreateAttachmentResult" element="tns:CreateAttachmentResponse"/>
  <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
</wsdl:message>
<wsdl:message name="DeleteAttachmentSoapIn" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:part name="request" element="tns:DeleteAttachment"/>
  <wsdl:part name="Impersonation" element="t:ExchangeImpersonation"/>
  <wsdl:part name="MailboxCulture" element="t:MailboxCulture"/>
  <wsdl:part name="RequestVersion" element="t:RequestServerVersion"/>
</wsdl:message>
<wsdl:message name="DeleteAttachmentSoapOut" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:part name="DeleteAttachmentResult" element="tns:DeleteAttachmentResponse"/>
</wsdl:message>
```
```xml
<soap:body parts="request" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="DeleteAttachmentResult" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
<soap:header message="tns:DeleteAttachmentSoapOut" part="ServerVersion" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetAttachment" xmlns:wsdl="http://schemas.xmlsoap.org/wsd1/">
xmlns:soap="http://schemas.xmlsoap.org/wsd1/">
<wsdl:input>
<soap:body parts="request" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="GetAttachmentResult" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
<soap:header message="tns:GetAttachmentSoapOut" part="ServerVersion" use="literal"
xmlns:soap="http://schemas.xmlsoap.org/wsd1/soap/*"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>
```
### Appendix B: Full XML Schema

For ease of implementation, the following sections provide the full XML schema for this protocol.

<table>
<thead>
<tr>
<th>Schema name</th>
<th>Prefix</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages schema</td>
<td>m:</td>
<td>7.1</td>
</tr>
<tr>
<td>Types schema</td>
<td>t:</td>
<td>7.2</td>
</tr>
</tbody>
</table>

These files have to be placed in a common folder in order for the WSDL to validate and operate. Also, any schema files that are included in or imported into the MS-OXWSATT-types.xsd or MS-OXWSATT-messages.xsd schemas have to be placed in the common folder along with the files listed in the table.

#### 7.1 Messages Schema

This section contains the contents of the MS-OXWSATT-messages.xsd file and information about additional files that this schema file requires to operate correctly.

MS-OXWSATT-messages.xsd includes the file listed in the following table. For the schema file to operate correctly, this file has to be present in the folder that contains the WSDL, types schema, and messages schema files for this protocol.

<table>
<thead>
<tr>
<th>File name</th>
<th>Defining specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-OXWSCDATA-messages.xsd</td>
<td>[MS-OXWSCDATA] section 7.1</td>
</tr>
</tbody>
</table>

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
  xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/messages"
  xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://schemas.microsoft.com/exchange/services/2006/messages"
  elementFormDefault="qualified" version="Exchange2016" id="messages">
  <xs:include schemaLocation="MS-OXWSCDATA-messages.xsd"/>
  <xs:import namespace="http://schemas.microsoft.com/exchange/services/2006/types"
                schemaLocation="MS-OXWSATT-types.xsd"/>
  <xs:complexType name="AttachmentInfoResponseMessageType">
    <xs:complexContent>
      <xs:extension base="m:ResponseMessageType">
        <xs:sequence>
          <xs:element name="Attachments" type="t:ArrayOfAttachmentsType"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="CreateAttachment" type="m:CreateAttachmentType"/>
  <xs:element name="CreateAttachmentResponse" type="m:CreateAttachmentResponseType"/>
  <xs:complexType name="CreateAttachmentResponseType">
    <xs:complexContent>
      <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="CreateAttachmentType">
    <xs:complexContent>
      <xs:extension base="m:BaseRequestType">
        <xs:sequence>
          <xs:element name="ParentItemId" type="t:ItemIdType"/>
          <xs:element name="Attachments" type="t:NonEmptyArrayOfAttachmentsType"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
```
7.2 Types Schema

This section contains the contents of the MS-OXWSATT-types.xsd file and information about additional files that this schema file requires to operate correctly.

MS-OXWSATT-types.xsd includes the files listed in the following table. For the schema file to operate correctly, these files have to be present in the folder that contains the WSDL, types schema, and messages schema files for this protocol.

<table>
<thead>
<tr>
<th>File name</th>
<th>Defining specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-OXWSCONT-types.xsd</td>
<td>[MS-OXWSCONT] section 7.2</td>
</tr>
<tr>
<td>MS-OXWSMSG-types.xsd</td>
<td>[MS-OXWSMSG] section 7</td>
</tr>
<tr>
<td>MS-OXWSMTGS-types.xsd</td>
<td>[MS-OXWSMTGS] section 7.2</td>
</tr>
<tr>
<td>MS-OXWSPOST-types.xsd</td>
<td>[MS-OXWSPOST] section 7.2</td>
</tr>
</tbody>
</table>

<?xml version="1.0" encoding="utf-8"?>
  <xs:include schemaLocation="MS-OXWSCONT-types.xsd"/>
  <xs:include schemaLocation="MS-OXWSMSG-types.xsd"/>
  <xs:include schemaLocation="MS-OXWSMTGS-types.xsd"/>
  <xs:include schemaLocation="MS-OXWSPOST-types.xsd"/>

  <xs:complexType name="ReferenceAttachmentType">
    <xs:complexContent>
      <xs:extension base="t:AttachmentType">
        <xs:sequence>
          <xs:element name="AttachLongPathName" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="ProviderType" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="ProviderEndpointUrl" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="AttachmentThumbnailUrl" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="AttachmentPreviewUrl" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="PermissionType" type="xs:int" minOccurs="0" maxOccurs="1"/>
          <xs:element name="AttachmentIsFolder" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="ArrayOfAttachmentsType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="ItemAttachment" type="t:ItemAttachmentType"/>
      <xs:element name="FileAttachment" type="t:FileAttachmentType"/>
      <xs:element name="ReferenceAttachment" type="t:ReferenceAttachmentType"/>
    </xs:choice>
  </xs:complexType>

  <xs:complexType name="AttachmentIdType">
    <xs:complexContent>
      <xs:extension base="t:RequestAttachmentIdType">
        <xs:attribute name="RootItemId" type="xs:string" use="optional"/>
        <xs:attribute name="RootItemChangeKey" type="xs:string" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="AttachmentResponseShapeType">
    <xs:sequence>
      <xs:element name="IncludeMimeContent" type="xs:boolean" minOccurs="0"/>
      <xs:element name="BodyType" type="t:BodyTypeResponseType" minOccurs="0"/>
      <xs:element name="FilterHtmlContent" type="xs:boolean" minOccurs="0"/>
      <xs:element name="AdditionalProperties" type="t:NonEmptyArrayOfPathsToElementType" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="AttachmentType">
    <xs:sequence>
      <xs:element name="AttachmentId" type="t:AttachmentIdType" minOccurs="0"/>  
      <xs:element name="Name" type="xs:string" minOccurs="0"/>  
      <xs:element name="ContentType" type="xs:string" minOccurs="0"/>  
      <xs:element name="ContentId" type="xs:string" minOccurs="0"/>  
      <xs:element name="ContentLocation" type="xs:string" minOccurs="0"/>  
      <xs:element name="Size" type="xs:int" minOccurs="0"/>  
      <xs:element name="LastModifiedTime" type="xs:dateTime" minOccurs="0"/>  
      <xs:element name="IsInline" type="xs:boolean" minOccurs="0"/>  
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="FileAttachmentType">
    <xs:complexContent>
      <xs:extension base="t:AttachmentType">
        <xs:sequence>
      </xs:sequence>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
8 Appendix C: Product Behavior

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

- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Microsoft Exchange Server 2019

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates 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 2.2.4.4: The IsInline element is not supported in Exchange 2007.

<2> Section 2.2.4.5: The IsContactPhoto element is not supported in Exchange 2007.

<3> Section 2.2.4.6: Exchange 2007 does not return the items of type t:ItemType as a t:MessageType type. Microsoft Exchange Server 2007 Service Pack 1 (SP1) starts to return the items of type t:ItemType as a t:MessageType type.

<4> Section 2.2.4.6: Exchange 2007, Exchange 2010, and Exchange 2013 do not support the RoleMember element.

<5> Section 2.2.4.6: Exchange 2007, Exchange 2010, and Exchange 2013 do not support the Network element.

<6> Section 2.2.4.6: Exchange 2007, Exchange 2010, and Exchange 2013 do not support the Person element.

<7> Section 2.2.4.6: Exchange 2007, Exchange 2010, and Exchange 2013 do not support the Booking element.

<8> Section 2.2.4.8: Exchange 2007, Exchange 2010, and the initial release of Exchange 2013 do not support the ReferenceAttachmentType complex type. This type was introduced in Microsoft Exchange Server 2013 Service Pack 1 (SP1).

<9> Section 3.1.4.3.3.3: The FilterHtmlContent element is not supported in Exchange 2007.

<10> Section 3.1.4.3.3.3: In Exchange 2007, Exchange 2010, Microsoft Exchange Server 2010 Service Pack 1 (SP1) and Microsoft Exchange Server 2010 Service Pack 2 (SP2), if the IncludeMimeContent element is set to true in the AttachmentResponseShapeType complex type, the MIME content will be returned for attachments of the following class: IPM.Note, IPM.Post, IPM.Appointment. If the IncludeMimeContent element is set to true and the attachment is not one of the accepted item classes, the GetAttachmentResponseMessage element MUST return an ErrorUnsupportedMimeConversion response code as specified in [MS-OXWSCDATA] section 2.2.5.24.
9 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.
## 10 Index

### A
- Abstract data model
  - Client 35
  - Server 21
- Applicability 9
- Attribute groups 20
- Attributes 20

### C
- Capability negotiation 9
- Change tracking 50
- Client
  - Abstract data model 35
    - Initialization 35
    - Local events 36
    - Message processing 35
    - Sequencing rules 35
    - Timer events 36
    - Timers 35
- Complex types 12
  - m:AttachmentInfoResponseMessageType Complex Type 13
  - t:ArrayOfAttachmentsType Complex Type 12
  - t:AttachmentIdType Complex Type 13
  - t:AttachmentType Complex Type 14
  - t:FileAttachmentType Complex Type 15
  - t:ItemAttachmentType Complex Type 16
  - t:NonEmptyArrayOfRequestAttachmentIdsType Complex Type 18
  - t:ReferenceAttachmentType Complex Type 18
  - t:RequestAttachmentIdType Complex Type 19
- Create attachment example 37

### D
- Data model - abstract
  - Client 35
  - Server 21
- Delete attachment example 38

### E
- Events
  - Local - Client 36
  - Local - Server 35
  - Timer - Client 36
  - Timer - Server 35
- Examples
  - Create attachment 37
  - Delete attachment 38
  - Get attachment 39
  - Overview 37

### F
- Fields - vendor-extensible 10
- Full WSDL 42
- Full XML schema 45
- Messages Schema 45
- Types Schema 46

### G
- Get attachment example 39
- Glossary 6
- Groups 20

### I
- Implementer - security considerations 41
- Index of security parameters 41
- Informative references 9
- Initialization
  - Client 35
  - Server 21
- Introduction 6

### L
- Local events
  - Client 36
  - Server 35

### M
- m:AttachmentInfoResponseMessageType Complex Type complex type 13
- Message processing
  - Client 35
  - Server 21
- Messages
  - Attribute groups 20
  - Attributes 20
  - Complex types 12
  - Elements 11
  - Enumerated 11
  - Groups 20
  - m:AttachmentInfoResponseMessageType Complex Type complex type 13
  - Namespaces 11
  - Simple types 20
  - Syntax 11
  - t:ArrayOfAttachmentsType Complex Type complex type 12
  - t:AttachmentIdType Complex Type complex type 13
  - t:AttachmentType Complex Type complex type 14
  - t:FileAttachmentType Complex Type complex type 15
  - t:ItemAttachmentType Complex Type complex type 16
  - t:NonEmptyArrayOfRequestAttachmentIdsType Complex Type complex type 18
  - t:ReferenceAttachmentType Complex Type complex type 18
  - t:RequestAttachmentIdType Complex Type complex type 19
  - Transport 11

### N
- Namespaces 11