

[MS-OXWSFOLD]: Folders and Folder Permissions Web Service Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: <http://www.microsoft.com/interop/osp>) or the Community Promise (available here: <http://www.microsoft.com/interop/cp/default.mspx>). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
07/15/2009	1.0	Major	Initial Availability.
11/04/2009	1.0.1	Editorial	Revised and edited the technical content.
02/10/2010	2.0.0	Major	Updated and revised the technical content.
05/05/2010	3.0.0	Major	Updated and revised the technical content.

Table of Contents

1 Introduction	7
1.1 Glossary	7
1.2 References	7
1.2.1 Normative References	7
1.2.2 Informative References	8
1.3 Overview	8
1.4 Relationship to Other Protocols	8
1.5 Prerequisites/Preconditions	9
1.6 Applicability Statement	9
1.7 Versioning and Capability Negotiation	9
1.8 Vendor-Extensible Fields	9
1.9 Standards Assignments	9
2 Messages	10
2.1 Transport	10
2.2 Common Message Syntax	10
2.2.1 Namespaces	10
2.2.2 Simple Types	10
2.2.2.1 t:FolderClassType Simple Type	11
2.2.2.2 t:PermissionActionType Simple Type	11
2.2.2.3 t:PermissionLevelType Simple Type	11
2.2.2.4 t:PermissionReadAccessType Simple Type	13
2.2.3 Complex Types	13
2.2.3.1 t:ArrayOfFoldersType Complex Type	14
2.2.3.2 m:BaseMoveCopyFolderType Complex Type	15
2.2.3.3 t:BasePermissionType Complex Type	15
2.2.3.4 t:ArrayOfUnknownEntriesType Complex Type	17
2.2.3.5 t:BaseFolderIdType Complex Type	17
2.2.3.6 t:BaseFolderType Complex Type	18
2.2.3.7 t:FolderChangeDescriptionType Complex Type	19
2.2.3.8 t:FolderChangeType Complex Type	20
2.2.3.9 m:FolderInfoResponseMessageType Complex Type	20
2.2.3.10 t:FolderType Complex Type	21
2.2.3.11 t:ManagedFolderInformationType Complex Type	22
2.2.3.12 t:PermissionSetType Complex Type	23
2.2.3.13 t:PermissionType Complex Type	23
2.2.3.14 t:TargetFolderIdType Complex Type	24
2.2.4 Elements	25
2.2.5 Attributes	25
2.2.6 Groups	25
2.2.7 Attribute Groups	25
2.2.8 Message Syntax	25
3 Protocol Details	26
3.1 ExchangeServicePortType Server Detail	26
3.1.1 Abstract Data Model	26
3.1.2 Timers	26
3.1.3 Initialization	26
3.1.4 Message Processing Events and Sequencing	26
3.1.4.1 CopyFolder	27

3.1.4.1.1	CopyFolder Simple Types	27
3.1.4.1.2	CopyFolder Complex Types.....	27
3.1.4.1.2.1	m:CopyFolderResponseType Complex Type	27
3.1.4.1.2.2	m:CopyFolderType Complex Type	28
3.1.4.1.3	CopyFolder Elements	28
3.1.4.1.3.1	CopyFolder Element.....	28
3.1.4.1.3.2	CopyFolderResponse Element.....	28
3.1.4.1.4	CopyFolder Attributes	29
3.1.4.1.5	CopyFolder Groups	29
3.1.4.1.6	CopyFolder Attribute Groups.....	29
3.1.4.1.7	CopyFolder Messages	29
3.1.4.1.7.1	tns:CopyFolderSoapIn.....	29
3.1.4.1.7.2	tns:CopyFolderSoapOut.....	29
3.1.4.2	CreateFolder	29
3.1.4.2.1	CreateFolder Simple Types	30
3.1.4.2.2	CreateFolder Complex Types	30
3.1.4.2.2.1	t:CreateFolderResponseType Complex Type	30
3.1.4.2.2.2	m:CreateFolderType Complex Type	31
3.1.4.2.2.3	t:NonEmptyArrayOfFoldersType Complex Type.....	31
3.1.4.2.3	CreateFolder Elements	32
3.1.4.2.3.1	CreateFolder Element.....	32
3.1.4.2.3.2	CreateFolderResponse Element	32
3.1.4.2.4	CreateFolder Attributes	32
3.1.4.2.5	CreateFolder Groups	33
3.1.4.2.6	CreateFolder Attribute Groups	33
3.1.4.2.7	CreateFolder Messages	33
3.1.4.2.7.1	tns:CreateFolderSoapIn.....	33
3.1.4.2.7.2	tns:CreateFolderSoapOut	33
3.1.4.3	CreateManagedFolder.....	33
3.1.4.3.1	CreateFolder Simple Types	34
3.1.4.3.2	CreateManagedFolder Complex Types	34
3.1.4.3.2.1	m:CreateManagedFolderRequestType Complex Type.....	34
3.1.4.3.2.2	m:CreateManagedFolderResponseType Complex Type.....	35
3.1.4.3.2.3	t:NonEmptyArrayOfFolderNamesType Complex Type.....	35
3.1.4.3.3	CreateManagedFolder Elements	36
3.1.4.3.3.1	CreateManagedFolder Element	36
3.1.4.3.3.2	CreateManagedFolderResponse Element.....	36
3.1.4.3.4	CreateManagedFolder Attributes	36
3.1.4.3.5	CreateManagedFolder Group.....	36
3.1.4.3.6	CreateManagedFolder Attribute Groups	36
3.1.4.3.7	CreateManagedFolder Messages.....	36
3.1.4.3.7.1	tns:CreateManagedFolderSoapIn	37
3.1.4.3.7.2	tns:CreateManagedFolderSoapOut	37
3.1.4.4	DeleteFolder	37
3.1.4.4.1	CreateManagedFolder Simple Types	38
3.1.4.4.2	DeleteFolder Complex Types.....	38
3.1.4.4.2.1	m>DeleteFolderResponseType Complex Type	38
3.1.4.4.2.2	m>DeleteFolderType Complex Type	38
3.1.4.4.3	DeleteFolder Elements	39
3.1.4.4.3.1	DeleteFolder Element	39
3.1.4.4.3.2	DeleteFolderResponse Element.....	39
3.1.4.4.4	DeleteFolder Attributes	39
3.1.4.4.5	DeleteFolder Groups	39

3.1.4.4.6	DeleteFolder Attribute Groups	40
3.1.4.4.7	DeleteFolder Messages	40
3.1.4.4.7.1	tns>DeleteFolderSoapIn	40
3.1.4.4.7.2	tns>DeleteFolderSoapOut	40
3.1.4.5	GetFolder	40
3.1.4.5.1	GetFolder Simple Types	41
3.1.4.5.2	GetFolder Complex Types	41
3.1.4.5.2.1	m:GetFolderResponseType Complex Type	41
3.1.4.5.2.2	m:GetFolderType Complex Type	41
3.1.4.5.2.3	t:NonEmptyArrayOfBaseFolderIdsType Complex Type	42
3.1.4.5.3	GetFolder Elements	42
3.1.4.5.3.1	GetFolder Element	43
3.1.4.5.3.2	GetFolderResponse Element	43
3.1.4.5.4	GetFolder Attributes	43
3.1.4.5.5	GetFolder Groups	43
3.1.4.5.6	GetFolder Attribute Groups	43
3.1.4.5.7	GetFolder Messages	43
3.1.4.5.7.1	tns:GetFolderSoapIn	43
3.1.4.5.7.2	tns:GetFolderSoapOut	44
3.1.4.6	MoveFolder	44
3.1.4.6.1	MoveFolder Simple Types	44
3.1.4.6.2	MoveFolder Complex Types	44
3.1.4.6.2.1	m:MoveFolderResponseType Complex Type	45
3.1.4.6.2.2	m:MoveFolderType Complex Type	45
3.1.4.6.3	MoveFolder Elements	45
3.1.4.6.3.1	MoveFolder Element	45
3.1.4.6.3.2	MoveFolderResponse Element	46
3.1.4.6.4	MoveFolder Attributes	46
3.1.4.6.5	MoveFolder Groups	46
3.1.4.6.6	MoveFolder Attribute Groups	46
3.1.4.6.7	MoveFolder Messages	46
3.1.4.6.7.1	tns:MoveFolderSoapIn	46
3.1.4.6.7.2	tns:MoveFolderSoapOut	46
3.1.4.7	UpdateFolder	47
3.1.4.7.1	UpdateFolder Simple Types	47
3.1.4.7.2	UpdateFolder Complex Types	47
3.1.4.7.2.1	m:UpdateFolderResponseType Complex Type	48
3.1.4.7.2.2	m:UpdateFolderType Complex Type	48
3.1.4.7.2.3	t:AppendToFolderFieldType Complex Type	48
3.1.4.7.2.4	t>DeleteFolderFieldType Complex Type	49
3.1.4.7.2.5	t:NonEmptyArrayOfFolderChangeDescriptionsType Complex Type	50
3.1.4.7.2.6	t:NonEmptyArrayOfFolderChangesType Complex Type	50
3.1.4.7.2.7	t:SetFolderFieldType Complex Type	51
3.1.4.7.3	UpdateFolder Elements	52
3.1.4.7.3.1	UpdateFolder Element	52
3.1.4.7.3.2	UpdateFolderResponse Element	52
3.1.4.7.4	UpdateFolder Attributes	52
3.1.4.7.5	UpdateFolder Groups	52
3.1.4.7.6	UpdateFolder Attribute Groups	52
3.1.4.7.7	UpdateFolder Messages	52
3.1.4.7.7.1	tns:UpdateFolderSoapIn	52
3.1.4.7.7.2	tns:UpdateFolderSoapOut	53
3.1.5	Server Timer Events	53

3.1.6	Server Other Local Events	53
3.2	Client Details.....	53
3.2.1	Abstract Data Model	53
3.2.2	Timers	53
3.2.3	Initialization	53
3.2.4	Message Processing Events and Sequencing	54
3.2.5	Timer Events	54
3.2.6	Other Local Events	54
4	Protocol Examples.....	55
4.1	CopyFolder Example.....	55
4.2	CreateFolder Example	56
4.3	DeleteFolder Example.....	57
4.4	MoveFolder Example	58
4.5	UpdateFolder Example.....	59
5	Security.....	61
5.1	Security Considerations for Implementers.....	61
5.2	Index of Security Parameters	61
6	Appendix A: Full WSDL	62
6.1	WSDL.....	62
6.2	Types Schema	67
6.3	Messages Schema.....	71
7	Appendix B: Product Behavior	74
8	Change Tracking.....	75
9	Index	77

1 Introduction

This document specifies the Folders and Folder Permissions Web Service Protocol Specification, which is used by clients to manipulate and organize **folders**, and retrieve and set permissions on a **Folder object**. Folder operations provide a way to manipulate folder and messages inside the folder.

1.1 Glossary

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

Active Directory
contact
folder
Folder object
Hypertext Transport Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
mailbox
message
search folder
SOAP message
special folder
store
task
Web Services Description Language (WSDL)
WSDL message
WSDL port type
XML
XML namespace
XML schema

The following terms are specific to this document:

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

1.2 References

1.2.1 Normative References

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

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

[MS-OXWSCDATA] Microsoft Corporation, "[Common Web Service Data Types](#)", July 2009.

[MS-OXWSXPROP] Microsoft Corporation, "[Extended Properties Web Service Schema](#)", July 2009.

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

[RFC3066] Alvestrand, H., "Tags for the Identification of Languages", RFC 3066, BCP 47, January 2001, <http://www.ietf.org/rfc/rfc3066.txt>

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", W3C Note, May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

[XMLNS] Bray, T., Hollander, D., Layman, A., Eds., et al., "Namespaces in XML 1.0 (Third Edition)", December 2009, <http://www.w3.org/TR/REC-xml-names/>

[XMLSCHEMA1] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>

[XMLSCHEMA2] Biron, P., and Malhotra, A., Eds., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>

1.2.2 Informative References

None.

1.3 Overview

The Folders and Folder Permissions Web Service Protocol provides clients with the folder operations for retrieving folder permission lists that are stored on the server. The protocol specifies the elements and operations for creating and locating the **special folders** in a **mailbox**.

1.4 Relationship to Other Protocols

The Folders and Folder Permissions Web Service protocol uses SOAP over **HTTP** and SOAP over **HTTPS**, as shown in the following figures.

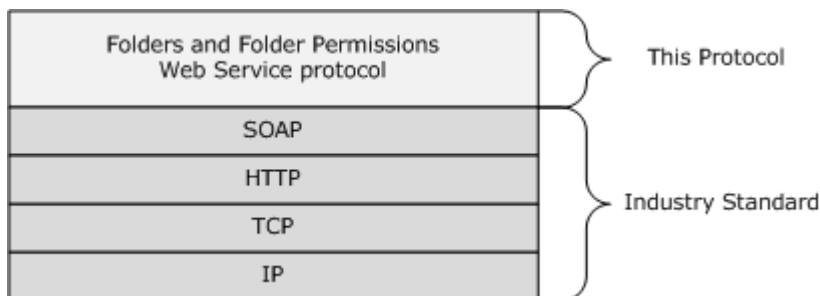


Figure 1: SOAP over HTTP

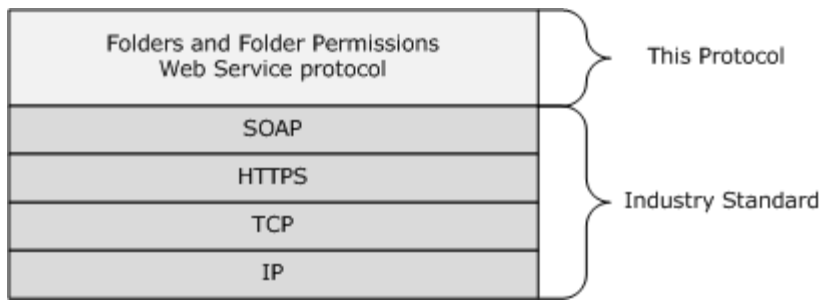


Figure 2: SOAP over HTTPS

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

The specification is applicable to environments that copy, create, delete, get, move, or update folders, using Exchange Web Services.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses SOAP 1.1, as specified in section [2.1](#).
- **Protocol Versions:** This protocol specifies only one **WSDL port type** version.
- **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 section [3.1.4](#).
- **Capability Negotiation:** None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The SOAP version supported is SOAP 1.1. For details, see [\[SOAP1.1\]](#).

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** 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.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
tns	http://schemas.microsoft.com/exchange/services/2006/messages	[MS-OXWSFOLD]
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1]
targetNamespace	http://schemas.microsoft.com/exchange/services/2006/messages	[MS-OXWSFOLD]
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
t	http://schemas.microsoft.com/exchange/services/2006/types	[MS-OXWSFOLD]

2.2.2 Simple Types

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

Simple Type	Description
t:FolderClassType	A string representation of the folder.
t:PermissionActionType	The PermissionActionType enumeration defines which items in a folder a user has permission to edit or delete.
t:PermissionLevelType	The PermissionLevelType enumeration specifies the permission level that a user has on a folder.
t:PermissionReadAccessType	The PermissionReadAccessType enumeration specifies whether a user has permission to read items in a folder.

2.2.2.1 t:FolderClassType Simple Type

The **FolderClassType** simple type represents the folder.

```
<xs:simpleType name="FolderClassType">
  <xs:restriction
    base="xs:string"
  />
</xs:simpleType>
```

2.2.2.2 t:PermissionActionType Simple Type

The **PermissionActionType** simple type enumeration defines which items in a folder a user has permission to edit or delete.

```
<xs:simpleType name="PermissionActionType">
  <xs:restriction
    base="xs:string"
  >
    <xs:enumeration
      value="None"
    />
    <xs:enumeration
      value="Owned"
    />
    <xs:enumeration
      value="All"
    />
  </xs:restriction>
</xs:simpleType>
```

Enumeration

The following values are defined by the **PermissionActionType** simple type:

Value	Description
None	Indicates that the user does not have permission to perform the action on any items in the folder.
Owned	Indicates that the user has permission to perform the action on the items in the folder that the user owns.
All	Indicates that the user has permission to perform the action on all items in the folder.

2.2.2.3 t:PermissionLevelType Simple Type

The **PermissionLevelType** simple type enumeration specifies the permission level that a user has on a folder.

```
<xs:simpleType name="PermissionLevelType">
  <xs:restriction
```

```

    base="xs:string"
  >
    <xs:enumeration
      value="None"
    />
    <xs:enumeration
      value="Owner"
    />
    <xs:enumeration
      value="PublishingEditor"
    />
    <xs:enumeration
      value="Editor"
    />
    <xs:enumeration
      value="PublishingAuthor"
    />
    <xs:enumeration
      value="Author"
    />
    <xs:enumeration
      value="NoneditingAuthor"
    />
    <xs:enumeration
      value="Reviewer"
    />
    <xs:enumeration
      value="Contributor"
    />
    <xs:enumeration
      value="Custom"
    />
  </xs:restriction>
</xs:simpleType>

```

Enumeration

The following values are defined by the **PermissionLevelType** simple type:

Value	Description
None	The user has no permissions on the folder.
Owner	The user can create, read, edit, and delete all items in the folder, and create subfolders. The user is both folder owner and folder contact.
PublishingEditor	The user can create, read, edit, and delete all items in the folder, and create subfolders.
Editor	The user can create, read, edit, and delete all items in the folder.
PublishingAuthor	The user can create and read all items in the folder, edit and delete only items that the user creates, and create subfolders.
Author	The user can create and read all items in the folder, and edit and delete only items that the user creates.

Value	Description
NoneditingAuthor	The user can create and read all items in the folder, and delete only items that the user creates.
Reviewer	The user can read all items in the folder.
Contributor	The user can create items in the folder. The contents of the folder do not appear.
Custom	The user has custom access permissions on the folder.

2.2.2.4 t:PermissionReadAccessType Simple Type

The **PermissionReadAccessType** simple type enumeration specifies whether a user has permission to read items in a folder.

```
<xs:simpleType name="PermissionReadAccessType">
  <xs:restriction
    base="xs:string"
  >
    <xs:enumeration
      value="None"
    />
    <xs:enumeration
      value="FullDetails"
    />
  </xs:restriction>
</xs:simpleType>
```

Enumeration

The following values are defined by the **PermissionReadAccessType** simple type:

Value	Description
None	The user does not have permission to read items in the folder.
FullDetails	The user has permission to read all items in the folder.

2.2.3 Complex Types

The following XML schema complex types are specific to this operation.

ComplexType	Description
ArrayOfFoldersType	An array of t:FolderTypes , which determines the type of folder.
BaseMoveCopyFolderType	Base type for the CopyFolderType and MoveFolderType types.
BasePermissionType	Base type that defines base permissions for items and folders.
ArrayOfUnknownEntriesType	An array of unknown permission entries that cannot be resolved against the Active Directory directory service.

ComplexType	Description
BaseFolderIdType	The base element for derived elements that represent a folder identifier.
BaseFolderType	The element class for derived elements that represent a folder.
FolderChangeDescriptionType	Describes a change to a single folder property .
FolderChangeType	Represents a collection of changes to be performed on a single folder.
FolderInfoResponseMessageType	Represents the response message.
FolderType	Represents a regular folder in the server database.
ManagedFolderInformationType	Contains information about a managed custom folder.
PermissionSetType	Contains all the permissions that are configured for a folder.
PermissionType	Specifies a permission on a folder.
TargetFolderIdType	Represents a target folder for operations that create, save, copy, move, or synchronize items or folder.

2.2.3.1 t:ArrayOfFoldersType Complex Type

The **ArrayOfFoldersType** complex type specifies an array of **FolderTypes** complex types (section [2.2.3.10](#)), which determines the type of folder.

```
<xs:complexType name="ArrayOfFoldersType">
  <xs:choice
    minOccurs="0"
    maxOccurs="unbounded"
  >
    <xs:element name="Folder"
      type="t:FolderType"
    />
    <xs:element name="CalendarFolder"
      type="t:CalendarFolderType"
    />
    <xs:element name="ContactsFolder"
      type="t:ContactsFolderType"
    />
    <xs:element name="SearchFolder"
      type="t:SearchFolderType"
    />
    <xs:element name="TasksFolder"
      type="t:TasksFolderType"
    />
  </xs:choice>
</xs:complexType>
```

Child Elements

Element	Type	Description
Folder	t:FolderType	Represents a regular folder in the server database.
CalendarFolder	t:CalendarFolderType	Represents a folder that primarily contains calendar items.
ContactsFolder	t:ContactsFolderType	Represents a contacts folder in a mailbox.
SearchFolder	t:SearchFolderType	Represents a search folder that is contained in a mailbox.
TasksFolder	t:TasksFolderType	Represents a task folder that is contained in a mailbox.

2.2.3.2 m:BaseMoveCopyFolderType Complex Type

The **BaseMoveCopyFolderType** complex type is the base type for the **CopyFolderType** (section [3.1.4.1.2.2](#)) and **MoveFolderType** (section [3.1.4.6.2.2](#)) complex types.

```
<xs:complexType name="BaseMoveCopyFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
    >
      <xs:sequence>
        <xs:element name="ToFolderId"
          type="t:TargetFolderIdType"
        />
        <xs:element name="FolderIds"
          type="t:NonEmptyArrayOfBaseFolderIdsType"
        />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Child Elements

Element	Type	Description
ToFolderId	t:TargetFolderIdType	Represents a target folder for operations that create, save, copy, move, or synchronize items or folders.
FolderIds	t:NonEmptyArrayOfBaseFolderIdsType	Specifies an array of one or more folders.

2.2.3.3 t:BasePermissionType Complex Type

The **BasePermissionType** complex type is an abstract type that defines base permissions for items and folders.

```
<xs:complexType name="BasePermissionType"
  abstract="true"
>
  <xs:sequence>
    <xs:element name="UserId"
```

```

        type="t:UserIdType"
        maxOccurs="1"
        minOccurs="1"
    />
    <xs:element name="CanCreateItems"
        type="xs:boolean"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="CanCreateSubFolders"
        type="xs:boolean"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="IsFolderOwner"
        type="xs:boolean"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="IsFolderVisible"
        type="xs:boolean"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="IsFolderContact"
        type="xs:boolean"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="EditItems"
        type="t:PermissionActionType"
        minOccurs="0"
        maxOccurs="1"
    />
    <xs:element name="DeleteItems"
        type="PermissionActionType"
        minOccurs="0"
        maxOccurs="1"
    />
</xs:sequence>
</xs:complexType>

```

Child Elements

Element	Type	Description
UserId	t:UserIdType	Specifies a user identifier.
CanCreateItems	xs:boolean	Specifies whether a client can create items in a folder.
CanCreateSubFolders	xs:boolean	Specifies whether the client can create subfolders.
IsFolderOwner	xs:boolean	Specifies whether the user is the owner of a folder.
IsFolderVisible	xs:boolean	Specifies whether a user can view a folder.

Element	Type	Description
IsFolderContact	xs:boolean	Specifies whether a user is a contact for a folder.
EditItems	t:PermissionActionType	Specifies whether a client can edit items.
DeleteItems	PermissionActionType	Specifies whether the client can delete items.

The **PermissionType** complex type ([\[MS-OXWSMTGS\]](#) section 2.2.3.13) and the **CalendarPermissionType** complex type ([\[MS-OXWSMTGS\]](#) section 2.2.3.6) extend the **BasePermissionType** complex type.

2.2.3.4 t:ArrayOfUnknownEntriesType Complex Type

The **ArrayOfUnknownEntriesType** complex type represents an array of unknown permission entries that cannot be resolved against the directory service. The text value represents a security identifier (SID).

```
<xs:complexType name="ArrayOfUnknownEntriesType">
  <xs:choice
    minOccurs="0"
    maxOccurs="unbounded"
  >
    <xs:element name="UnknownEntry"
      type="xs:string"
    />
  </xs:choice>
</xs:complexType>
```

Child Elements

Element	Type	Description
UnknownEntry	xs:string	Represents an array of unknown permission entries that cannot be resolved against the directory service.

2.2.3.5 t:BaseFolderIdType Complex Type

The **BaseFolderIdType** complex type specifies the base element for derived elements that represent a folder identifier. The **BaseFolderIdType** complex type MUST NOT be sent in a SOAP message because it is an abstract type.

```
<xs:complexType name="BaseFolderIdType"
  abstract="true"
/>
```

The **DistinguishedFolderIdType** complex type ([\[MS-OXWSCDATA\]](#) section 2.2.3.21) and the **FolderIdType** complex type ([\[MS-OXWSCDATA\]](#) section 2.2.3.29) extend the **BaseFolderIdType** complex type.

2.2.3.6 t:BaseFolderType Complex Type

The **BaseFolderType** complex type specifies the base type for folders. The **BaseFolderType** complex type **MUST NOT** be sent in a **SOAP message** because it is an abstract type.

```
<xs:complexType name="BaseFolderType"
  abstract="true"
>
  <xs:sequence>
    <xs:element name="FolderId"
      type="FolderIdType"
      minOccurs="0"
    />
    <xs:element name="ParentFolderId"
      type="FolderIdType"
      minOccurs="0"
    />
    <xs:element name="FolderClass"
      type="xs:string"
      minOccurs="0"
    />
    <xs:element name="DisplayName"
      type="xs:string"
      minOccurs="0"
    />
    <xs:element name="TotalCount"
      type="xs:int"
      minOccurs="0"
    />
    <xs:element name="ChildFolderCount"
      type="xs:int"
      minOccurs="0"
    />
    <xs:element name="ExtendedProperty"
      type="t:ExtendedPropertyType"
      minOccurs="0"
      maxOccurs="unbounded"
    />
    <xs:element name="ManagedFolderInformation"
      type="ManagedFolderInformationType"
      minOccurs="0"
    />
    <xs:element name="EffectiveRights"
      type="EffectiveRightsType"
      minOccurs="0"
    />
  </xs:sequence>
</xs:complexType>
```

Child Elements

Element	Type	Description
FolderId	FolderIdType	Specifies the folder identifier and change key.

Element	Type	Description
ParentFolderId	FolderIdType	Specifies the parent folder identifier.
FolderClass	xs:string	Specifies the folder class. This value MUST be "IPF.Appointment" for calendar folders, "IPF.Contact" for contact folders, "IPF.Journal" for journal folders, "IPF.Note" for mail folders, "IPF.StickyNote" for note folders, and "IPF.Task" for task folders. The folders class can be a custom class. The following example shows how to set a custom folder class: "IPF.<folderclass>", where <folderclass> is the name of the custom class.
DisplayName	xs:string	Specifies the display name of the folder.
TotalCount	xs:int	Specifies the total number of items in a folder. This property MUST be read-only for a client and is only returned in a response.
ChildFolderCount	xs:int	Specifies the total number of child folders in a folder. This property MUST be read-only for a client and is only returned in a response.
ExtendedProperty	t:ExtendedPropertyType	Specifies the set of extended properties on a folder.
ManagedFolderInformation	ManagedFolderInformationType	Specifies metadata for a managed folder. This property MUST be read-only for a client and is only returned in a response. This complex type is specified in section 2.2.3.11 .
EffectiveRights	EffectiveRightsType	Specifies the operations that the current logon user can perform on the folder. This property MUST be read-only for a client and is only returned in a response.

The **FolderType** complex type (section [2.2.3.10](#)), **CalendarFolderType** complex type ([\[MS-OXWSMTGS\]](#) section 2.2.3.3), and **ContactsFolderType** complex type ([\[MS-OXWSCONT\]](#) section 3.1.4.1.2.2) extend the **BaseFolderType** complex type.

2.2.3.7 t:FolderChangeDescriptionType Complex Type

The **FolderChangeDescriptionType** complex type specifies a change to a single folder property.

```
<xs:complexType name="FolderChangeDescriptionType">
  <xs:complexContent>
    <xs:extension
      base="ChangeDescriptionType"
```

```

    />
  </xs:complexContent>
</xs:complexType>

```

The **SetFolderFieldType** complex type (section [3.1.4.7.2.7](#)), the **DeleteFolderFieldType** complex type (section [3.1.4.7.2.4](#)), and the **AppendToFolderFieldType** complex type (section [3.1.4.7.2.3](#)) extend the **FolderChangeDescriptionType** complex type.

2.2.3.8 t:FolderChangeType Complex Type

The **FolderChangeType** complex type specifies a collection of changes to be performed on a single folder.

```

<xs:complexType name="FolderChangeType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="FolderId"
        type="t:FolderIdType"
      />
      <xs:element name="DistinguishedFolderId"
        type="t:DistinguishedFolderIdType"
      />
    </xs:choice>
    <xs:element name="Updates"
      type="t:NonEmptyArrayOfFolderChangeDescriptionsType"
    />
  </xs:sequence>
</xs:complexType>

```

Child Elements

Element	Type	Description
FolderId	t:FolderIdType	Specifies an identifier for a fully resolved folder.
DistinguishedFolderId	t:DistinguishedFolderIdType	Specifies an identifier for a distinguished folder.
Updates	t:NonEmptyArrayOfFolderChangeDescriptionsType	Specifies a collection of changes to a folder.

2.2.3.9 m:FolderInfoResponseMessageType Complex Type

The **FolderInfoResponseMessageType** complex type represents the response message for the **CreateFolder** operation (section [3.1.4.2](#)), **GetFolder** operation (section [3.1.4.5](#)), **UpdateFolder** operation (section [3.1.4.7](#)), **MoveFolder** operation (section [3.1.4.6](#)), **CopyFolder** operation (section [3.1.4.1](#)), and **CreateManagedFolder** operation (section [3.1.4.3](#)).

```

<xs:complexType name="FolderInfoResponseMessageType">

```

```

<xs:complexContent>
  <xs:extension
    base="m:ResponseMessageType"
  >
    <xs:sequence>
      <xs:element name="Folders"
        type="t:ArrayOfFoldersType"
        minOccurs="0"
      />
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type	Description
Folders	t:ArrayOfFoldersType	Represents the response message.

2.2.3.10 t:FolderType Complex Type

The **FolderType** complex type represents a regular folder in the server database.

```

<xs:complexType name="FolderType">
  <xs:complexContent>
    <xs:extension
      base="t:BaseFolderType"
    >
      <xs:sequence>
        <xs:element name="PermissionSet"
          type="t:PermissionSetType"
          minOccurs="0"
        />
        <xs:element name="UnreadCount"
          type="xs:int"
          minOccurs="0"
        />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type	Description
PermissionSet	t:PermissionSetType	Specifies all permissions that are configured for a folder.
UnreadCount	xs:int	Specifies the number of unread items in a folder. This element MUST only exist in responses and MUST equal the sum of all MessageType and PostItemType item types that have the IsRead property set to false .

The **SearchFolderType** ([\[MS-OXWSSRCH\]](#) section 2.2.3.26) and the **TasksFolderType** ([\[MS-OXWSTASK\]](#) section 2.2.3.2) extend the **FolderType** complex type.

2.2.3.11 t:ManagedFolderInformationType Complex Type

The **ManagedFolderInformationType** complex type contains information about a managed custom folder.

```
<xs:complexType name="ManagedFolderInformationType">
  <xs:sequence>
    <xs:element name="CanDelete"
      type="xs:boolean"
      minOccurs="0"
    />
    <xs:element name="CanRenameOrMove"
      type="xs:boolean"
      minOccurs="0"
    />
    <xs:element name="MustDisplayComment"
      type="xs:boolean"
      minOccurs="0"
    />
    <xs:element name="HasQuota"
      type="xs:boolean"
      minOccurs="0"
    />
    <xs:element name="IsManagedFoldersRoot"
      type="xs:boolean"
      minOccurs="0"
    />
    <xs:element name="ManagedFolderId"
      type="xs:string"
      minOccurs="0"
    />
    <xs:element name="Comment"
      type="xs:string"
      minOccurs="0"
    />
    <xs:element name="StorageQuota"
      type="xs:int"
      minOccurs="0"
    />
    <xs:element name="FolderSize"
      type="xs:int"
      minOccurs="0"
    />
    <xs:element name="HomePage"
      type="xs:string"
      minOccurs="0"
    />
  </xs:sequence>
</xs:complexType>
```

Child Elements

Element	Type	Description
CanDelete	xs:boolean	A value that indicates whether a managed folder can be deleted by a client.
CanRenameOrMove	xs:boolean	A value that indicates whether a managed folder can be renamed or moved by a client.
MustDisplayComment	xs:boolean	A value that indicates whether the managed folder comment must be displayed.
HasQuota	xs:boolean	A value that indicates whether a managed folder has a quota.
IsManagedFoldersRoot	xs:boolean	A value that indicates whether the managed folder is the root managed folder.
ManagedFolderId	xs:string	The unique identifier of a managed folder.
Comment	xs:string	A comment that is associated with a managed folder.
StorageQuota	xs:int	The storage quota for a managed folder.
FolderSize	xs:int	A value that describes the total size of all the contents of a managed folder.
HomePage	xs:string	The URL that is the default home page for the managed folder.

2.2.3.12 t:PermissionSetType Complex Type

The **PermissionSetType** complex type contains all the permissions that are configured for a folder. This constitutes the set of permissions on a folder.

```
<xs:complexType name="PermissionSetType">
  <xs:sequence>
    <xs:element name="Permissions"
      type="t:ArrayOfPermissionsType"
    />
    <xs:element name="UnknownEntries"
      type="t:ArrayOfUnknownEntriesType"
      minOccurs="0"
    />
  </xs:sequence>
</xs:complexType>
```

Child Elements

Element	Type
Permissions	t:ArrayOfPermissionsType
UnknownEntries	t:ArrayOfUnknownEntriesType

2.2.3.13 t:PermissionType Complex Type

The **PermissionType** complex type specifies a permission on a folder.

```

<xs:complexType>
  <xs:complexContent>
    <xs:extension
      base="t:BasePermissionType"
    >
      <xs:sequence>
        <xs:element name="ReadItems"
          type="t:PermissionReadAccessType"
          minOccurs="0"
        />
        <xs:element name="PermissionLevel"
          type="t:PermissionLevelType"
        />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type	Description
ReadItems	t:PermissionReadAccessType	Specifies whether a user has permission to read items in a folder.
PermissionLevel	t:PermissionLevelType	Specifies the combination of permissions that a user has on a folder.

This type defines the access that a user has to a folder.

2.2.3.14 t:TargetFolderIdType Complex Type

The **TargetFolderIdType** complex type specifies a target folder for operations that create, save, copy, move, or synchronize items or folders.

```

<xs:complexType name="TargetFolderIdType">
  <xs:choice>
    <xs:element name="FolderId"
      type="t:FolderIdType"
    />
    <xs:element name="DistinguishedFolderId"
      type="t:DistinguishedFolderIdType"
    />
  </xs:choice>
</xs:complexType>

```

Child Elements

Element	Type	Description
FolderId	t:FolderIdType	Specifies a folder identifier.
DistinguishedFolderId	t:DistinguishedFolderIdType	Specifies a distinguished folder identifier.

This type is a container for either a **FolderIdType** complex type (section [2.2.3.29](#)) or **DistinguishedFolderIdType** complex type ([\[MS-OXWSCDATA\]](#) section 2.2.3.21) object.

2.2.4 Elements

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

2.2.5 Attributes

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

2.2.6 Groups

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

2.2.7 Attribute Groups

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

2.2.8 Message Syntax

This specification does not define any common XML schema message 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 Detail

The Folders and Folder Permissions Web Service Protocol has seven operations to manipulate folders.

Operation	Description
CreateFolder	Creates folders, calendar folders, contacts folders, tasks folders, and search folder.
DeleteFolder	Deletes folders from a mailbox.
UpdateFolder	Modifies properties of an existing item in the server store.
MoveFolder	Moves folders from a specified folder and puts them in another folder.
CopyFolder	Copies identified folders and returns the Id and ChangeKey of the copied folders.
GetFolder	Gets folders from the server store .
CreateManagedFolder	Creates a managed folder in the server store.

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.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing

This protocol includes the operations listed in the following table.

Operation	Description
CreateFolder	Creates folders, calendar folders, contacts folders, tasks folders, and search folders.
DeleteFolder	Deletes folders from a mailbox.
UpdateFolder	Modifies properties of an existing item in the server store.

Operation	Description
MoveFolder	Moves folders from a specified folder and puts them in another folder.
CopyFolder	Copies identified folders and returns the Id and ChangeKey of the copied folders.
GetFolder	Gets folders from the server store.
CreateManagedFolder	Creates a managed folder in the server store.

3.1.4.1 CopyFolder

This protocol uses the **CopyFolder** operation to copy identified folders and returns the Id and ChangeKey of the copied folders.

```
<wsdl:operation name="CopyFolder">
  <wsdl:input message="tns:CopyFolderSoapIn" />
  <wsdl:output message="tns:CopyFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns:CopyFolderSoapIn	Specifies the SOAP message that copies the folders.

Response

Message Format	Description
tns:CopyFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.1.1 CopyFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.1.2 CopyFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:CopyFolderResponseType	Represents a response to a CopyFolder operation.
m:CopyFolderType	Represents an operation to copy folders in a server database.

3.1.4.1.2.1 m:CopyFolderResponseType Complex Type

The **CopyFolderResponseType** complex type represents a response to a **CopyFolder** operation (section [3.1.4.1](#)).

```
<xs:complexType name="CopyFolderResponseType">
```

```

<xs:complexContent>
  <xs:extension
    base="m:BaseResponseMessageType"
  />
</xs:complexContent>
</xs:complexType>

```

3.1.4.1.2.2 m:CopyFolderType Complex Type

The **CopyFolderType** complex type represents an operation to copy folders in a server database. The **CopyFolderType** complex type extends the **BaseMoveCopyFolderType** as specified in section [2.2.3.2](#).

```

<xs:complexType name="CopyFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseMoveCopyFolderType"
    />
  </xs:complexContent>
</xs:complexType>

```

3.1.4.1.3 CopyFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns:CopyFolder	Defines a request to copy folders in a mailbox in the server store.
tns:CopyFolderResponse	Defines a response to a CopyFolder request.

3.1.4.1.3.1 CopyFolder Element

The <CopyFolder> element defines a request to copy folders in a mailbox in the server store.

```

<xs:element name="CopyFolder"
  type="m:CopyFolderType"
/>

```

3.1.4.1.3.2 CopyFolderResponse Element

The <CopyFolderResponse> element defines a response to a **CopyFolder** operation (section [3.1.4.1](#)).

```

<xs:element name="CopyFolderResponse"
  type="m:CopyFolderResponseType"
/>

```

3.1.4.1.4 CopyFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.1.5 CopyFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.1.6 CopyFolder Attribute Groups

This specification does not define any specific XML schema attribute group definitions for this operation.

3.1.4.1.7 CopyFolder Messages

The following **WSDL message** definitions are specific to this operation.

Messages	Description
tns:CopyFolderSoapIn	The request that copies a folder.
tns:CopyFolderSoapOut	The response from a copy folder request.

3.1.4.1.7.1 tns:CopyFolderSoapIn

The **CopyFolderSoapIn** message contains four parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:CopyFolder	Specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	Specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .
RequestVersion	t:RequestServerVersion	Specifies the schema version for the CopyFolder operation (section 3.1.4.1) request.

3.1.4.1.7.2 tns:CopyFolderSoapOut

The **CopyFolderSoapOut** message contains two parts, as described in the following table.

Part Name	Element/Type	Description
CopyFolderResult	tns:CopyFolderResponse	Specifies the response.
ServerVersion	t:ServerVersionInfo	Specifies the server version for the response.

3.1.4.2 CreateFolder

This protocol uses the **CreateFolder** operation to create folders, calendar folders, contacts folders, tasks folders, and search folders.

Before any data can be read from or written to a folder, an implementation needs to ensure that the folder exists, and open or create it, if it does not exist. Before a folder can be created, the parent folder **MUST** already exist. Trying to create a folder that already exists results in an error.

```
<wsdl:operation name="CreateFolder">
  <wsdl:input message="tns:CreateFolderSoapIn" />
  <wsdl:output message="tns:CreateFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns:CreateFolderSoapIn	Specifies the SOAP message that creates the folders.

Response

Message Format	Description
tns:CreateFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.2.1 CreateFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.2.2 CreateFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:CreateFolderResponseType	Represents a response to a CreateFolder operation.
m:CreateFolderType	Represents an operation to create folders in a server database.
t:NonEmptyArrayOfFoldersType	An array of folders that has at least one member.

3.1.4.2.2.1 t:CreateFolderResponseType Complex Type

The **CreateFolderResponseType** complex type represents a response to a **CreateFolder** operation (section [3.1.4.2.2](#)).

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

3.1.4.2.2.2 m:CreateFolderType Complex Type

The **CreateFolderType** complex type represents a request to create a folder.

```
<xs:complexType name="CreateFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
    >
      <xs:sequence>
        <xs:element name="ParentFolderId"
          type="t:TargetFolderIdType"
        />
        <xs:element name="Folders"
          type="t:NonEmptyArrayOfFoldersType"
        />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Child Elements

Element	Type	Description
ParentFolderId	t:TargetFolderIdType	The identifier of the folder that will contain the newly created folder.
Folders	t:NonEmptyArrayOfFoldersType	Represents a target folder for operations that create, save, copy, move, or synchronize items or folder.

3.1.4.2.2.3 t:NonEmptyArrayOfFoldersType Complex Type

The **NonEmptyArrayOfFoldersType** complex type represents an array of folders that has at least one member.

```
<xs:complexType name="NonEmptyArrayOfFoldersType">
  <xs:choice
    maxOccurs="unbounded"
  >
    <xs:element name="Folder"
      type="t:FolderType"
    />
    <xs:element name="CalendarFolder"
      type="t:CalendarFolderType"
    />
    <xs:element name="ContactsFolder"
      type="t:ContactsFolderType"
    />
    <xs:element name="SearchFolder"
      type="t:SearchFolderType"
    />
    <xs:element name="TasksFolder"
      type="t:TasksFolderType"
    />
  </xs:choice>
</xs:complexType>
```

```

    />
  </xs:choice>
</xs:complexType>

```

Child Elements

Element	Type	Description
Folder	t:FolderType	Represents a regular folder in the server database.
CalendarFolder	t:CalendarFolderType	Represents a folder that primarily contains calendar items.
ContactsFolder	t:ContactsFolderType	Represents a Contacts folder in a mailbox.
SearchFolder	t:SearchFolderType	Represents a search folder that is contained in a mailbox.
TasksFolder	t:TasksFolderType	Represents a Tasks folder that is contained in a mailbox.

3.1.4.2.3 CreateFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns:CreateFolder	Defines a request to create folders in a mailbox in the server store.
tns:CreateFolderResponse	Defines a response to a CreateFolder request.

3.1.4.2.3.1 CreateFolder Element

The <CreateFolder> element defines a request to create a folder in a mailbox in the server store.

```

<xs:element name="CreateFolder"
  type="m:CreateFolderType"
/>

```

3.1.4.2.3.2 CreateFolderResponse Element

The <CreateFolderResponse> element defines a response to a **CreateFolder** operation (section [3.1.4.2](#)) request.

```

<xs:element name="CreateFolderResponse"
  type="m:CreateFolderResponseType"
/>

```

3.1.4.2.4 CreateFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.2.5 CreateFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.2.6 CreateFolder Attribute Groups

This specification does not define any specific XML schema attribute group definitions for this operation.

3.1.4.2.7 CreateFolder Messages

The following WSDL message definitions are specific to this operation.

Messages	Description
tns:CreateFolderSoapIn	The request that creates a folder.
tns:CreateFolderSoapOut	The response from a create folder request.

3.1.4.2.7.1 tns:CreateFolderSoapIn

The [CreateFolderSoapIn](#) message contains five parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:CreateFolder	This part specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	This part specifies the culture to use for accessing the server. The cultures are defined by RFC3066 .
RequestVersion	t:RequestServerVersion	This part specifies the schema version for the CreateFolder request.
TimeZoneContext	t:TimeZoneContext	Specifies a time zone definition and enables associating SOAP attributes with the definition.

3.1.4.2.7.2 tns:CreateFolderSoapOut

The [CreateFolderSoapOut](#) message contains two parts, as described in the following table.

Part Name	Element/Type	Description
CreateFolderResult	tns:CreateFolderResponse	This part specifies the response.
ServerVersion	t:ServerVersionInfo	This part specifies the server version for the response.

3.1.4.3 CreateManagedFolder

This protocol uses the **CreateManagedFolder** operation to create a managed folder in the server store.

```
<wsdl:operation name="CreateManagedFolder">
```

```

    <wsdl:input message="tns:CreateManagedFolderSoapIn" />
    <wsdl:output message="tns:CreateManagedFolderSoapOut" />
</wsdl:operation>

```

Request

Message Format	Description
tns:CreateManagedFolderSoapIn	Specifies the SOAP message that creates the managed folder.

Response

Message Format	Description
tns:CreateManagedFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

A managed folder is created by an administrator and placed in a user's mailbox for messaging records management purposes. The retention and journaling of messages in managed folders are controlled by managed content settings that are applied to the folder.

3.1.4.3.1 CreateFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.3.2 CreateManagedFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:CreateManagedFolderRequestType	Represents an operation to create a managed folder in a server database.
m:CreateManagedFolderResponseType	Represents a response to a CreateManagedFolder operation.
t:NonEmptyArrayOfFolderNamesType	An array of named managed folders in a mailbox that has at least one member.

3.1.4.3.2.1 m:CreateManagedFolderRequestType Complex Type

The **CreateManagedFolderRequestType** complex type represents an operation to create a managed folder in a server database.

```

<xs:complexType name="CreateManagedFolderRequestType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
    >
      <xs:sequence>
        <xs:element name="FolderNames"
          type="t:NonEmptyArrayOfFolderNamesType"
        />
      </xs:sequence>
    </xs:complexContent>
  </xs:complexType>

```

```

    <xs:element name="Mailbox"
      type="t:EmailAddressType"
      minOccurs="0"
    />
  </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type	Description
FolderNames	t:NonEmptyArrayOfFolderNamesType	An array of managed folders to add to a mailbox. This array has at least one member.
Mailbox	t:EmailAddressType	The e-mail address of the mailbox in which the managed folders are added.

3.1.4.3.2.2 m:CreateManagedFolderResponseType Complex Type

The **CreateManagedFolderResponseType** complex type represents a response to a **CreateManagedFolder** operation (section [3.1.4.3](#)).

```

<xs:complexType name="CreateManagedFolderResponseType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseResponseMessageType"
    />
  </xs:complexContent>
</xs:complexType>

```

3.1.4.3.2.3 t:NonEmptyArrayOfFolderNamesType Complex Type

The **NonEmptyArrayOfFolderNamesType** complex type represents an array of named managed folders in a mailbox that has at least one member.

```

<xs:complexType name="NonEmptyArrayOfFolderNamesType">
  <xs:sequence>
    <xs:element name="FolderName"
      type="xs:string"
      maxOccurs="unbounded"
    />
  </xs:sequence>
</xs:complexType>

```

Child Elements

Element	Type
FolderName	xs:string

3.1.4.3.3 CreateManagedFolder Elements

The following XML Schema element definitions are specific to this operation.

Operation	Description
tns:CreateManagedFolder	Defines a request to add a managed folder to a mailbox in the server store.
tns:CreateManagedFolderResponse	Defines a response to a CreateManagedFolder request.

3.1.4.3.3.1 CreateManagedFolder Element

The <CreateManagedFolder> element defines a request to create a managed folder in a mailbox in the server store.

```
<xs:element name="CreateManagedFolder"
  type="m:CreateManagedFolderRequestType"
  />
```

3.1.4.3.3.2 CreateManagedFolderResponse Element

The <CreateManagedFolderResponse> element defines a response to a **CreateManagedFolder** operation (section [3.1.4.3](#)) request.

```
<xs:element name="CreateManagedFolderResponse"
  type="m:CreateManagedFolderResponseType"
  />
```

3.1.4.3.4 CreateManagedFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.3.5 CreateManagedFolder Group

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.3.6 CreateManagedFolder Attribute Groups

This specification does not define any specific XML schema attribute group definitions for this operation.

3.1.4.3.7 CreateManagedFolder Messages

The following **WSDL message** definitions are specific to this operation.

Messages	Description
tns:CreateManagedFolderSoapIn	The request that creates a managed folder.
tns:CreateManagedFolderSoapOut	The response from a create managed folder request.

3.1.4.3.7.1 tns:CreateManagedFolderSoapIn

The [CreateManagedFolderSoapIn](#) message contains four parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:CreateManagedFolder	This part specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	This part specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .
RequestVersion	t:RequestServerVersion	This part specifies the schema version for the CreateManagedFolder request.

3.1.4.3.7.2 tns:CreateManagedFolderSoapOut

The [CreateManagedFolderSoapOut](#) message contains two parts, as described in the following table.

Part Name	Element/Type	Description
CreateManagedFolderResult	tns:CreateManagedFolderResponse	This part specifies the response.
ServerVersion	t:ServerVersionInfo	This part specifies the server version for the response.

3.1.4.4 DeleteFolder

This protocol uses the **DeleteFolder** operation to delete folders from a mailbox. The **DeleteFolder** operation is used to delete unmanaged folders and managed folders. You cannot delete default folders, such as the Inbox or Deleted Items folders, by using this operation. To be deleted, a folder MUST exist.

```
<wsdl:operation name="DeleteFolder">
  <wsdl:input message="tns:DeleteFolderSoapIn" />
  <wsdl:output message="tns:DeleteFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns>DeleteFolderSoapIn	Specifies the SOAP message that deletes a folder.

Response

Message Format	Description
tns:DeleteFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.4.1 CreateManagedFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.4.2 DeleteFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:DeleteFolderResponseType	Represents an operation to delete a folder in a server database.
m:DeleteFolderType	Represents a response to a DeleteFolder operation.

3.1.4.4.2.1 m:DeleteFolderResponseType Complex Type

The **DeleteFolderResponseType** complex type represents a response to a **DeleteFolder** operation (section [3.1.4.4](#)).

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

3.1.4.4.2.2 m:DeleteFolderType Complex Type

The **DeleteFolderType** complex type represents a request to delete folders from a mailbox.

```
<xs:complexType name="DeleteFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
    >
    <xs:sequence>
      <xs:element name="FolderIds"
        type="t:NonEmptyArrayOfBaseFolderIdsType"
      />
    </xs:sequence>
    <xs:attribute name="DeleteType"
      type="t:DisposalType"
      use="required"
    />
  </xs:extension>
</xs:complexType>
```

Child Elements

Element	Type
FolderIds	t:NonEmptyArrayOfBaseFolderIdsType

Attributes

Name	Type
DeleteType	t:DisposalType

3.1.4.4.3 DeleteFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns>DeleteFolder	Defines a request to delete a folder from a mailbox in the server store.
tns>DeleteFolderResponse	Defines a response to a DeleteFolder request.

3.1.4.4.3.1 DeleteFolder Element

The <DeleteFolder> element defines a request to delete a folder from a mailbox in the server store.

```
<xs:element name="DeleteFolder"
  type="m:DeleteFolderType"
/>
```

3.1.4.4.3.2 DeleteFolderResponse Element

The <DeleteFolderResponse> element represents a response to a **DeleteFolder** operation (section [3.1.4.4](#)).

```
<xs:element name="DeleteFolderResponse"
  type="m:DeleteFolderResponseType"
/>
```

3.1.4.4.4 DeleteFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.4.5 DeleteFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.4.6 DeleteFolder Attribute Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.4.7 DeleteFolder Messages

The following WSDL message definitions are specific to this operation.

Messages	Description
tns:DeleteFolderSoapIn	The request that deletes a folder.
tns:DeleteFolderSoapOut	The response from a delete folder request.

3.1.4.4.7.1 tns:DeleteFolderSoapIn

The **DeleteFolderSoapIn** message contains four parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:DeleteFolder	Specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	Specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .
RequestVersion	t:RequestServerVersion	Specifies the schema version for the DeleteFolder operation (section 3.1.4.4) request.

3.1.4.4.7.2 tns:DeleteFolderSoapOut

The **DeleteFolderSoapOut** message contains two parts, as described in the following table.

Part Name	Element/Type	Description
DeleteFolderResult	tns:DeleteFolderResponse	Specifies the response.
ServerVersion	t:ServerVersionInfo	Specifies the server version for the response.

3.1.4.5 GetFolder

This protocol uses the **GetFolder** operation to get folders, calendar folders, contacts folders, tasks folders, and search folders.

```
<wsdl:operation name="GetFolder">
  <wsdl:input message="tns:GetFolderSoapIn" />
  <wsdl:output message="tns:GetFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns:GetFolderSoapIn	Specifies the SOAP message that gets folders from the server store.

Response

Message Format	Description
tns:GetFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.5.1 GetFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.5.2 GetFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:GetFolderResponseType	Represents a response to a GetFolder operation.
m:GetFolderType	Represents an operation to get a folder in a server database.
t:NonEmptyArrayOfBaseFolderIdsType	Specifies an array of one or more folders.

3.1.4.5.2.1 m:GetFolderResponseType Complex Type

The **GetFolderResponseType** complex type represents a response to a **GetFolder** operation (section [3.1.4.5](#)).

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

3.1.4.5.2.2 m:GetFolderType Complex Type

The **GetFolderType** complex type represents an operation to get a folder in a server database.

```
<xs:complexType name="GetFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
      >
    <xs:sequence>
      <xs:element name="FolderShape"
        type="t:FolderResponseShapeType"
        />
    </xs:sequence>
  </xs:complexContent>
</xs:complexType>
```

```

    <xs:element name="FolderIds"
      type="t:NonEmptyArrayOfBaseFolderIdsType"
    />
  </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type
FolderShape	t:FolderResponseShapeType
FolderIds	t:NonEmptyArrayOfBaseFolderIdsType

3.1.4.5.2.3 t:NonEmptyArrayOfBaseFolderIdsType Complex Type

The **NonEmptyArrayOfBaseFolderIdsType** complex type specifies an array of one or more folders.

```

<xs:complexType>
  <xs:choice
    maxOccurs="unbounded"
  >
    <xs:element name="FolderId"
      type="t:FolderIdType"
    />
    <xs:element name="DistinguishedFolderId"
      type="t:DistinguishedFolderIdType"
    />
  </xs:choice>
</xs:complexType>

```

Child Elements

Element	Type	Description
FolderId	t:FolderIdType	Identifies a folder.
DistinguishedFolderId	t:DistinguishedFolderIdType	Specifies a distinguished folder identifier.

3.1.4.5.3 GetFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns:GetFolder	Defines a request to get a folder from a mailbox in the server store.
tns:GetFolderResponse	Defines a response to a GetFolder request.

3.1.4.5.3.1 GetFolder Element

The <GetFolder> element defines a request to get a folder from a mailbox in the server store.

```
<xs:element name="GetFolder"
  type="m:GetFolderType"
/>
```

3.1.4.5.3.2 GetFolderResponse Element

The <GetFolderResponse> element represents a response to a **GetFolder** operation (section [3.1.4.5](#)) request.

```
<xs:element name="GetFolderResponse"
  type="m:GetFolderResponseType"
/>
```

3.1.4.5.4 GetFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.5.5 GetFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.5.6 GetFolder Attribute Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.5.7 GetFolder Messages

The following WSDL message definitions are specific to this operation.

Messages	Description
tns:GetFolderSoapIn	The request that gets a folder.
tns:GetFolderSoapOut	The response from a get folder request.

3.1.4.5.7.1 tns:GetFolderSoapIn

The **GetFolderSoapIn** message contains five parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:GetFolder	Specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	Specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .

Part Name	Element/Type	Description
RequestVersion	t:RequestServerVersion	Specifies the schema version for the GetFolder operation (section 3.1.4.5) request.
TimeZoneContext	t:TimeZoneContext	Specifies a time zone definition and enables associating SOAP attributes with the definition.

3.1.4.5.7.2 tns:GetFolderSoapOut

The **GetFolderSoapOut** message contains two parts, as described in the following table.

Part Name	Element/Type	Description
GetFolderResult	tns:GetFolderResponse	Specifies the response message.
ServerVersion	t:ServerVersionInfo	Specifies the server version for the response.

3.1.4.6 MoveFolder

This protocol uses the **MoveFolder** operation to move folders from a specified parent folder and puts them in another parent folder. All the properties, contents, and subfolders of the folder are moved with the folder.

```
<wsdl:operation name="MoveFolder">
  <wsdl:input message="tns:MoveFolderSoapIn" />
  <wsdl:output message="tns:MoveFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns:MoveFolderSoapIn	Specifies the SOAP message that moves folders from a specified folder and puts them in another folder.

Response

Message Format	Description
tns:MoveFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.6.1 MoveFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.6.2 MoveFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:MoveFolderResponseType	Represents a response to a MoveFolder operation.
m:MoveFolderType	Represents an operation to move a folder in a mailbox.

3.1.4.6.2.1 m:MoveFolderResponseType Complex Type

The **MoveFolderResponseType** complex type represents a response to a **MoveFolder** operation (section [3.1.4.6](#)).

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

3.1.4.6.2.2 m:MoveFolderType Complex Type

The **MoveFolderType** complex type represents an operation to move folders in a mailbox. The **MoveFolderType** complex type extends the **BaseMoveCopyFolderType** as specified in section [2.2.3.2](#).

```
<xs:complexType name="MoveFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseMoveCopyFolderType"
    />
  </xs:complexContent>
</xs:complexType>
```

3.1.4.6.3 MoveFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns:MoveFolder	Defines a request to move a folder in the server store.
tns:MoveFolderResponse	Defines a response to a MoveFolder request.

3.1.4.6.3.1 MoveFolder Element

The <MoveFolder> element defines a request to move a folder in the server store.

```
<xs:element name="MoveFolder"
  type="m:MoveFolderType"
/>
```

3.1.4.6.3.2 MoveFolderResponse Element

The **MoveFolderResponse** element represents a response to a [MoveFolder](#) request.

```
<xs:element name="MoveFolderResponse"
  type="m:MoveFolderResponseType"
/>
```

3.1.4.6.4 MoveFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.6.5 MoveFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.6.6 MoveFolder Attribute Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.6.7 MoveFolder Messages

The following WSDL message definitions are specific to this operation.

Messages	Description
tns:MoveFolderSoapIn	The request that moves a folder.
tns:MoveFolderSoapOut	The response from a move folder request.

3.1.4.6.7.1 tns:MoveFolderSoapIn

The [MoveFolderSoapIn](#) message contains four parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:MoveFolder	This part specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	This part specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .
RequestVersion	t:RequestServerVersion	This part specifies the schema version for the MoveFolder request.

3.1.4.6.7.2 tns:MoveFolderSoapOut

The [MoveFolderSoapOut](#) message contains two parts, as described in the following table.

Part Name	Element/Type	Description
MoveFolderResult	tns:MoveFolderResponse	This part specifies the response message.
ServerVersion	t:ServerVersionInfo	This part specifies the server version for the response.

3.1.4.7 UpdateFolder

This protocol uses the **UpdateFolder** operation to modify properties of an existing folder in the server store.

```
<wsdl:operation name="UpdateFolder">
  <wsdl:input message="tns:UpdateFolderSoapIn" />
  <wsdl:output message="tns:UpdateFolderSoapOut" />
</wsdl:operation>
```

Request

Message Format	Description
tns:UpdateFolderSoapIn	Specifies the SOAP message that requests the properties of an existing item in the server store.

Response

Message Format	Description
tns:UpdateFolderSoapOut	Specifies the SOAP message that is returned by the server in response.

3.1.4.7.1 UpdateFolder Simple Types

This specification does not define any specific XML schema simple types for this operation.

3.1.4.7.2 UpdateFolder Complex Types

The following XML schema complex types are specific to this operation.

Type	Description
m:UpdateFolderResponseType	Represents a response to an UpdateFolder operation.
m:UpdateFolderType	Represents a request to update folders in a mailbox.
t:AppendToFolderFieldType	Do not use. This complex type is not implemented.
t>DeleteFolderFieldType	Represents an UpdateFolder operation to delete a property from a folder.
t:NonEmptyArrayOfFolderChangeDescriptionsType	An array of t:ChangeDescriptionTypes that has at least one member.

Type	Description
t:NonEmptyArrayOfFolderChangesType	An array of t:FolderChangeTypes that has at least one member.
t:SetFolderFieldType	Represents an UpdateFolder operation to set a property on an existing folder.

3.1.4.7.2.1 m:UpdateFolderResponseType Complex Type

The **UpdateFolderResponseType** complex type represents a response to an [UpdateFolder](#) operation.

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

3.1.4.7.2.2 m:UpdateFolderType Complex Type

The **UpdateFolderType** complex type represents an operation to update folders in a mailbox.

```
<xs:complexType name="UpdateFolderType">
  <xs:complexContent>
    <xs:extension
      base="m:BaseRequestType"
    >
    <xs:sequence>
      <xs:element name="FolderChanges"
        type="t:NonEmptyArrayOfFolderChangesType"
      />
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>
```

Child Elements

Element	Type
FolderChanges	t:NonEmptyArrayOfFolderChangesType

3.1.4.7.2.3 t:AppendToFolderFieldType Complex Type

The **AppendToFolderFieldType** complex type is not implemented. Any request that uses this complex type will always return an error response.

```
<xs:complexType name="AppendToFolderFieldType">
```



```

<xs:complexContent>
  <xs:extension
    base="t:FolderChangeDescriptionType"
  >
    <xs:sequence>
      <xs:choice>
        <xs:element name="Folder"
          type="t:FolderType"
        />
        <xs:element name="CalendarFolder"
          type="t:CalendarFolderType"
        />
        <xs:element name="ContactsFolder"
          type="t:ContactsFolderType"
        />
        <xs:element name="SearchFolder"
          type="t:SearchFolderType"
        />
        <xs:element name="TasksFolder"
          type="t:TasksFolderType"
        />
      </xs:choice>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>

```

Child Elements

Element	Type	Description
Folder	t:FolderType	Represents a regular folder in the server database.
CalendarFolder	t:CalendarFolderType	Rrepresents a folder that primarily contains calendar items.
ContactsFolder	t:ContactsFolderType	Represents a Contacts folder in a mailbox.
SearchFolder	t:SearchFolderType	Represents a search folder that is contained in a mailbox.
TasksFolder	t:TasksFolderType	Represents a Tasks folder that is contained in a mailbox.

3.1.4.7.2.4 t:DeleteFolderFieldType Complex Type

The **DeleteFolderFieldType** complex type represents an [UpdateFolder](#) operation to delete a property from a folder.

```

<xs:complexType name="DeleteFolderFieldType">
  <xs:complexContent>
    <xs:extension
      base="t:FolderChangeDescriptionType"
    >
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

3.1.4.7.2.5 t:NonEmptyArrayOfFolderChangeDescriptionsType Complex Type

The **NonEmptyArrayOfFolderChangeDescriptionsType** complex type is an array of [t:ChangeDescriptionTypes](#) that has at least one member. The [t:FolderChangeDescriptionType](#) members describe a change to a single folder property.

```
<xs:complexType name="NonEmptyArrayOfFolderChangeDescriptionsType">
  <xs:choice
    maxOccurs="unbounded"
  >
    <xs:element name="AppendToFolderField"
      type="t:AppendToFolderFieldType"
    />
    <xs:element name="SetFolderField"
      type="t:SetFolderFieldType"
    />
    <xs:element name="DeleteFolderField"
      type="t>DeleteFolderFieldType"
    />
  </xs:choice>
</xs:complexType>
```

Child Elements

Element	Type	Description
AppendToFolderField	t:AppendToFolderFieldType	Do not use. This complex type is not implemented.
SetFolderField	t:SetFolderFieldType	Represents an UpdateFolder operation to set a property on an existing folder.
DeleteFolderField	t>DeleteFolderFieldType	Represents an UpdateFolder operation to delete a property from a folder.

3.1.4.7.2.6 t:NonEmptyArrayOfFolderChangesType Complex Type

The **NonEmptyArrayOfFolderChangesType** complex type represents an array of [t:FolderChangeTypes](#) that has at least one member. The **t:FolderChangeType** type represents a collection of changes to be performed on a single folder.

```
<xs:complexType name="NonEmptyArrayOfFolderChangesType">
  <xs:sequence>
    <xs:element name="FolderChange"
      type="t:FolderChangeType"
      maxOccurs="unbounded"
    />
  </xs:sequence>
</xs:complexType>
```

Child Elements

Element	Type	Description
FolderChange	t:FolderChangeType	Represents a collection of changes to be performed on a single folder.

In the [UpdateFolder Example](#), the folder request XML shows a FolderChanges type with a t:FolderChangeType complex type. The FolderId is identified and the changes are shown in the Updates element.

3.1.4.7.2.7 t:SetFolderFieldType Complex Type

The **SetFolderFieldType** complex type represents an [UpdateFolder](#) operation to set a property on an existing folder.

```
<xs:complexType name="SetFolderFieldType">
  <xs:complexContent>
    <xs:extension
      base="t:FolderChangeDescriptionType"
    >
      <xs:choice>
        <xs:element name="Folder"
          type="t:FolderType"
        />
        <xs:element name="CalendarFolder"
          type="t:CalendarFolderType"
        />
        <xs:element name="ContactsFolder"
          type="t:ContactsFolderType"
        />
        <xs:element name="SearchFolder"
          type="t:SearchFolderType"
        />
        <xs:element name="TasksFolder"
          type="t:TasksFolderType"
        />
      </xs:choice>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Child Elements

Element	Type	Description
Folder	t:FolderType	Represents a regular folder in the server database.
CalendarFolder	t:CalendarFolderType	Represents a folder that primarily contains calendar items.
ContactsFolder	t:ContactsFolderType	Represents a Contacts folder in a mailbox.
SearchFolder	t:SearchFolderType	Represents a search folder that is contained in a mailbox.
TasksFolder	t:TasksFolderType	Represents a Tasks folder that is contained in a mailbox.

3.1.4.7.3 UpdateFolder Elements

The following XML schema element definitions are specific to this operation.

Operation	Description
tns:UpdateFolder	Defines a request to update a folder in the server store.
tns:UpdateFolderResponse	Defines a response to an UpdateFolder request.

3.1.4.7.3.1 UpdateFolder Element

The <UpdateFolder> element defines a request to update a folder in the server store.

```
<xs:element name="UpdateFolder"
  type="m:UpdateFolderType"
/>
```

3.1.4.7.3.2 UpdateFolderResponse Element

The <UpdateFolderResponse> element represents a response to an [UpdateFolder](#) request.

```
<xs:element name="UpdateFolderResponse"
  type="m:UpdateFolderResponseType"
/>
```

3.1.4.7.4 UpdateFolder Attributes

This specification does not define any specific XML schema attribute definitions for this operation.

3.1.4.7.5 UpdateFolder Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.7.6 UpdateFolder Attribute Groups

This specification does not define any specific XML schema group definitions for this operation.

3.1.4.7.7 UpdateFolder Messages

The following WSDL message definitions are specific to this operation.

Messages	Description
tns:UpdateFolderSoapIn	The request that updates a folder.
tns:UpdateFolderSoapOut	The response from an update folder request.

3.1.4.7.7.1 tns:UpdateFolderSoapIn

The [UpdateFolderSoapIn](#) message contains five parts, as described in the following table.

Part Name	Element/Type	Description
request	tns:UpdateFolder	This part specifies the request.
Impersonation	t:ExchangeImpersonation	The identifier of the account to impersonate.
MailboxCulture	t:MailboxCulture	This part specifies the culture to use for accessing the server. The cultures are defined by [RFC3066] .
RequestVersion	t:RequestServerVersion	This part specifies the schema version for the UpdateFolder request.
TimeZoneContext	t:TimeZoneContext	Specifies a time zone definition and enables associating SOAP attributes with the definition.

3.1.4.7.2 tns:UpdateFolderSoapOut

The [MoveFolderSoapOut](#) message contains two parts, as described in the following table.

Part Name	Element/Type	Description
UpdateFolderResult	tns:UpdateFolderResponse	This part specifies the response message.
ServerVersion	t:ServerVersionInfo	This part specifies the server version for the response.

3.1.5 Server Timer Events

None.

3.1.6 Server 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 that are 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

None.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

4 Protocol Examples

The following examples, [CopyFolder Example](#), [CreateFolder Example](#), [DeleteFolder Example](#), [MoveFolder Example](#), and [UpdateFolder Example](#) show the request and response XML to perform the specific operations.

4.1 CopyFolder Example

The following is an example of a [CopyFolder](#) Operation with the [CopyFolderType](#) complex type.

The client constructs the request XML and sends it to the server. The identified folder, along with all contents, are copied to the destination folder, in this example, the Junk E-mail folder. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```
<?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:CopyFolder>
        <m:ToFolderId>
          <t:DistinguishedFolderId Id="junkemail" />
        </m:ToFolderId>
        <m:FolderIds>
          <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA" />
        </m:FolderIds>
      </m:CopyFolder>
    </soap:Body>
  </soap:Envelope>
```

The server constructs the response XML and sends it to the client. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```
<?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:CopyFolderResponse
        xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
        xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
        <m:ResponseMessages>
```

```

    <m:CopyFolderResponseMessage ResponseClass="Success">
      <m:ResponseCode>NoError</m:ResponseCode>
      <m:Folders>
        <t:Folder>
          <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA" />
        </t:Folder>
      </m:Folders>
    </m:CopyFolderResponseMessage>
  </m:ResponseMessages>
</m:CopyFolderResponse>
</s:Body>
</s:Envelope>

```

4.2 CreateFolder Example

The following is an example of a [CreateFolder](#) Operation with the [CreateFolderType](#) complex type.

The client constructs the request XML and sends it to the server. A new folder, by the name of Custom Folder is created in the inbox. If the newly created folder already exists within the specified parent folder, an error is thrown.

```

<?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:CreateFolder>
      <m:ParentFolderId>
        <t:DistinguishedFolderId Id="inbox" />
      </m:ParentFolderId>
      <m:Folders>
        <t:Folder>
          <t:FolderClass>IPF.MyCustomFolderClass</t:FolderClass>
          <t:DisplayName>Custom Folder</t:DisplayName>
          <t:PermissionSet>
            <t:Permissions />
          </t:PermissionSet>
        </t:Folder>
      </m:Folders>
    </m:CreateFolder>
  </soap:Body>
</soap:Envelope>

```

The server constructs the response XML and sends it to the client. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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:CreateFolderResponse
xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
        <m:ResponseMessages>
            <m:CreateFolderResponseMessage ResponseClass="Success">
                <m:ResponseCode>NoError</m:ResponseCode>
                <m:Folders>
                    <t:Folder>
                        <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA " />
                    </t:Folder>
                </m:Folders>
            </m:CreateFolderResponseMessage>
        </m:ResponseMessages>
    </m:CreateFolderResponse>
</s:Body>
</s:Envelope>

```

4.3 DeleteFolder Example

The following is an example of a [DeleteFolder](#) Operation with the [DeleteFolderType](#) complex type.

The client constructs the request XML and sends it to the server. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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>DeleteFolder DeleteType="SoftDelete">
            <m:FolderIds>
                <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABQAAAA" />
            </m:FolderIds>
        </m>DeleteFolder>
    </soap:Body>
</soap:Envelope>

```

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

```

<?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:DeleteFolderResponse
    xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
    xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
    <m:ResponseMessages>
      <m:DeleteFolderResponseMessage ResponseClass="Success">
        <m:ResponseCode>NoError</m:ResponseCode>
      </m:DeleteFolderResponseMessage>
    </m:ResponseMessages>
  </m:DeleteFolderResponse>
</s:Body>
</s:Envelope>

```

4.4 MoveFolder Example

The following is an example of a [MoveFolder](#) Operation with the [MoveFolderType](#) complex type. The specified folder is from its current parent folder to the sentitems folder.

The client constructs the request XML and sends it to the server. The specified folder is moved to the specified parent folder. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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:MoveFolder>
      <m:ToFolderId>
        <t:DistinguishedFolderId Id="sentitems" />
      </m:ToFolderId>
      <m:FolderIds>
        <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA " />
      </m:FolderIds>
    </m:MoveFolder>
  </soap:Body>
</soap:Envelope>

```

The server constructs the response XML and sends it to the client. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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:MoveFolderResponse
        xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
        xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
        <m:ResponseMessages>
          <m:MoveFolderResponseMessage ResponseClass="Success">
            <m:ResponseCode>NoError</m:ResponseCode>
            <m:Folders>
              <t:Folder>
                <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA " />
              </t:Folder>
            </m:Folders>
          </m:MoveFolderResponseMessage>
        </m:ResponseMessages>
      </m:MoveFolderResponse>
    </s:Body>
  </s:Envelope>

```

4.5 UpdateFolder Example

The following is an example of a [UpdateFolder](#) Operation with the [UpdateFolderType](#) complex type. This example modifies the DisplayName of the identified folder to "Modified Custom Folder".

The client constructs the xml and sends it to the server. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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:UpdateFolder>
      <m:FolderChanges>
        <t:FolderChange>
          <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABQAAAA" />
          <t:Updates>
            <t:SetFolderField>
              <t:FieldURI FieldURI="folder:DisplayName" />
              <t:Folder>
                <t:DisplayName>Modified Custom Folder</t:DisplayName>
              </t:Folder>
            </t:Updates>
          </t:FolderChange>
        </m:FolderChanges>
      </m:UpdateFolder>
    </soap:Body>
  </soap:Envelope>

```

```

        </t:SetFolderField>
    </t:Updates>
</t:FolderChange>
</m:FolderChanges>
</m:UpdateFolder>
</soap:Body>
</soap:Envelope>

```

The server constructs the response XML and sends it to the client. The FolderId Id and ChangeKey attributes have been shortened to preserve readability.

```

<?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:UpdateFolderResponse
      xmlns:m="http://schemas.microsoft.com/exchange/services/2006/messages"
      xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types">
      <m:ResponseMessages>
        <m:UpdateFolderResponseMessage ResponseClass="Success">
          <m:ResponseCode>NoError</m:ResponseCode>
          <m:Folders>
            <t:Folder>
              <t:FolderId Id="AAMkAGIwODEy" ChangeKey="AQAAABYAAA" />
            </t:Folder>
          </m:Folders>
        </m:UpdateFolderResponseMessage>
      </m:ResponseMessages>
    </m:UpdateFolderResponse>
  </s:Body>
</s:Envelope>ap:Envelope>

```

5 Security

5.1 Security Considerations for Implementers

The Folders and Folder Permissions Web service does not use additional security mechanisms.

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. The contents of each file are included in this section.

Section	File name	Description
WSDL	MS-OXWSFOLD.wsdl	Contains the WSDL for the implementation of this protocol.
Types Schema	MS-OXWSFOLD-types.xsd	Contains the XML schema type definitions that are used in this protocol.
Messages Schema	MS-OXWSFOLD-messages.xsd	Contains the XML schema message definitions that are used in this protocol.

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

6.1 WSDL

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

```
<?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:s="http://www.w3.org/2001/XMLSchema"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
targetNamespace="http://schemas.microsoft.com/exchange/services/2006/messages">
  <wSDL:types>
    <xs:schema id="messages"
      elementFormDefault="qualified"
      version="Exchange2010"
      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"
      xmlns="http://schemas.microsoft.com/exchange/services/2006/messages">
      <xs:include schemaLocation="MS-OXWSFOLD-messages.xsd"/>
      <!-- Add global elements and types from messages.xsd -->
    </xs:schema>
    <xs:schema id="types"
      elementFormDefault="qualified"
      version="Exchange2010"
      xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
      targetNamespace="http://schemas.microsoft.com/exchange/services/2006/types"
      xmlns="http://schemas.microsoft.com/exchange/services/2006/types"
      xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/types"
      xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
    </xs:schema>
  </wSDL:types>
  <wSDL:portType name="ExchangeServicePortType">
```

```

<wsdl:operation name="CreateFolder">
  <wsdl:input message="tns:CreateFolderSoapIn" />
  <wsdl:output message="tns:CreateFolderSoapOut" />
</wsdl:operation>
<wsdl:operation name="DeleteFolder">
  <wsdl:input message="tns>DeleteFolderSoapIn" />
  <wsdl:output message="tns>DeleteFolderSoapOut" />
</wsdl:operation>
<wsdl:operation name="UpdateFolder">
  <wsdl:input message="tns:UpdateFolderSoapIn" />
  <wsdl:output message="tns:UpdateFolderSoapOut" />
</wsdl:operation>
<wsdl:operation name="MoveFolder">
  <wsdl:input message="tns:MoveFolderSoapIn" />
  <wsdl:output message="tns:MoveFolderSoapOut" />
</wsdl:operation>
<wsdl:operation name="CopyFolder">
  <wsdl:input message="tns:CopyFolderSoapIn" />
  <wsdl:output message="tns:CopyFolderSoapOut" />
</wsdl:operation>
  <wsdl:operation name="GetFolder">
    <wsdl:input message="tns:GetFolderSoapIn" />
    <wsdl:output message="tns:GetFolderSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="CreateManagedFolder">
    <wsdl:input message="tns:CreateManagedFolderSoapIn" />
    <wsdl:output message="tns:CreateManagedFolderSoapOut" />
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ExchangeServiceBinding" type="tns:ExchangeServicePortType">
  <wsdl:documentation>
    <wsi:Claim conformsTo="http://ws-i.org/profiles/basic/1.0"
xmlns:wsi="http://ws-i.org/schemas/conformanceClaim/" />
  </wsdl:documentation>
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/GetFolder" />
    <wsdl:input>
      <soap:header message="tns:GetFolderSoapIn" part="Impersonation"
use="literal"/>
      <soap:header message="tns:GetFolderSoapIn" part="MailboxCulture"
use="literal"/>
      <soap:header message="tns:GetFolderSoapIn" part="RequestVersion"
use="literal"/>
      <soap:header message="tns:GetFolderSoapIn" part="TimeZoneContext"
use="literal"/>
      <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body parts="GetFolderResult" use="literal" />
      <soap:header message="tns:GetFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="CreateFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/CreateFolder" />
    <wsdl:input>

```

```

        <soap:header message="tns:CreateFolderSoapIn" part="Impersonation"
use="literal"/>
        <soap:header message="tns:CreateFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns:CreateFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:header message="tns:CreateFolderSoapIn" part="TimeZoneContext"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="CreateFolderResult" use="literal" />
        <soap:header message="tns:CreateFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DeleteFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/DeleteFolder" />
    <wsdl:input>
        <soap:header message="tns>DeleteFolderSoapIn" part="Impersonation"
use="literal"/>
        <soap:header message="tns>DeleteFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns>DeleteFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="DeleteFolderResult" use="literal" />
        <soap:header message="tns>DeleteFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="UpdateFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/UpdateFolder" />
    <wsdl:input>
        <soap:header message="tns:UpdateFolderSoapIn" part="Impersonation"
use="literal"/>
        <soap:header message="tns:UpdateFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns:UpdateFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:header message="tns:UpdateFolderSoapIn" part="TimeZoneContext"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="UpdateFolderResult" use="literal" />
        <soap:header message="tns:UpdateFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="MoveFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/MoveFolder" />
    <wsdl:input>
        <soap:header message="tns:MoveFolderSoapIn" part="Impersonation"
use="literal"/>

```



```

        <soap:header message="tns:MoveFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns:MoveFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="MoveFolderResult" use="literal" />
        <soap:header message="tns:MoveFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="CopyFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/CopyFolder" />
    <wsdl:input>
        <soap:header message="tns:CopyFolderSoapIn" part="Impersonation"
use="literal"/>
        <soap:header message="tns:CopyFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns:CopyFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="CopyFolderResult" use="literal" />
        <soap:header message="tns:CopyFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>
    <wsdl:operation name="CreateManagedFolder">
    <soap:operation
soapAction="http://schemas.microsoft.com/exchange/services/2006/messages/CreateManagedFolder"
/>
    </>
    <wsdl:input>
        <soap:header message="tns:CreateManagedFolderSoapIn" part="Impersonation"
use="literal"/>
        <soap:header message="tns:CreateManagedFolderSoapIn" part="MailboxCulture"
use="literal"/>
        <soap:header message="tns:CreateManagedFolderSoapIn" part="RequestVersion"
use="literal"/>
        <soap:body parts="request" use="literal" />
    </wsdl:input>
    <wsdl:output>
        <soap:body parts="CreateManagedFolderResult" use="literal" />
        <soap:header message="tns:CreateManagedFolderSoapOut" part="ServerVersion"
use="literal"/>
    </wsdl:output>
</wsdl:operation>

</wsdl:binding>
<wsdl:message name="GetFolderSoapIn">
    <wsdl:part name="request" element="tns:GetFolder" />
    <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="GetFolderSoapOut">

```

```

        <wsdl:part name="GetFolderResult" element="tns:GetFolderResponse" />
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="CreateFolderSoapIn">
        <wsdl:part name="request" element="tns:CreateFolder" />
        <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="CreateFolderSoapOut">
        <wsdl:part name="CreateFolderResult" element="tns:CreateFolderResponse" />
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="CreateManagedFolderSoapIn">
        <wsdl:part name="request" element="tns:CreateManagedFolder" />
        <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="CreateManagedFolderSoapOut">
        <wsdl:part name="CreateManagedFolderResult" element="tns:CreateManagedFolderResponse"
/>
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="DeleteFolderSoapIn">
        <wsdl:part name="request" element="tns>DeleteFolder" />
        <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="DeleteFolderSoapOut">
        <wsdl:part name="DeleteFolderResult" element="tns>DeleteFolderResponse" />
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="UpdateFolderSoapIn">
        <wsdl:part name="request" element="tns:UpdateFolder" />
        <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="UpdateFolderSoapOut">
        <wsdl:part name="UpdateFolderResult" element="tns:UpdateFolderResponse" />
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="MoveFolderSoapIn">
        <wsdl:part name="request" element="tns:MoveFolder" />
        <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="MoveFolderSoapOut">
        <wsdl:part name="MoveFolderResult" element="tns:MoveFolderResponse" />
        <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
    </wsdl:message>
    <wsdl:message name="CopyFolderSoapIn">
        <wsdl:part name="request" element="tns:CopyFolder" />
        <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="CopyFolderSoapOut">
    <wsdl:part name="CopyFolderResult" element="tns:CopyFolderResponse" />
    <wsdl:part name="ServerVersion" element="t:ServerVersionInfo"/>
  </wsdl:message>
</wsdl:definitions>

```

6.2 Types Schema

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

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

Defining specification	File name
[MS-OXWSCDATA] , section 6.3	MS-OXWSCDATA-types.xsd
[MS-OXWSXPROP] , section 6.1	MS-OXWSXPROP-types.xsd

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:tns="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://schemas.microsoft.com/exchange/services/2006/types"
  elementFormDefault="qualified" version="Exchange2010" id="types">
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
  <xs:include schemaLocation="MS-OXWSCDATA-types.xsd"/>
  <xs:include schemaLocation="MS-OXWSXPROP-types.xsd"/>
  <xs:complexType name="AppendToFolderFieldType">
    <xs:complexContent>
      <xs:extension base="t:FolderChangeDescriptionType">
        <xs:sequence>
          <xs:choice>
            <xs:element name="Folder" type="t:FolderType"/>
            <xs:element name="CalendarFolder" type="t:CalendarFolderType"/>
            <xs:element name="ContactsFolder" type="t:ContactsFolderType"/>
            <xs:element name="SearchFolder" type="t:SearchFolderType"/>
            <xs:element name="TasksFolder" type="t:TasksFolderType"/>
          </xs:choice>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ArrayOfFoldersType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="Folder" type="t:FolderType"/>
      <xs:element name="CalendarFolder" type="t:CalendarFolderType"/>
      <xs:element name="ContactsFolder" type="t:ContactsFolderType"/>
      <xs:element name="SearchFolder" type="t:SearchFolderType"/>
      <xs:element name="TasksFolder" type="t:TasksFolderType"/>
    </xs:choice>
  </xs:complexType>

```

```

</xs:complexType>
<xs:complexType name="ArrayOfUnknownEntriesType">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="UnknownEntry" type="xs:string"/>
  </xs:choice>
</xs:complexType>
<xs:complexType name="BaseFolderIdType" abstract="true">
</xs:complexType>
<xs:complexType name="BaseFolderType" abstract="true">
  <xs:sequence>
    <xs:element name="FolderId" type="t:FolderIdType" minOccurs="0"/>
    <xs:element name="ParentFolderId" type="t:FolderIdType" minOccurs="0"/>
    <xs:element name="FolderClass" type="xs:string" minOccurs="0"/>
    <xs:element name="DisplayName" type="xs:string" minOccurs="0"/>
    <xs:element name="TotalCount" type="xs:int" minOccurs="0"/>
    <xs:element name="ChildFolderCount" type="xs:int" minOccurs="0"/>
    <xs:element name="ExtendedProperty" type="t:ExtendedPropertyType"
minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="ManagedFolderInformation"
type="t:ManagedFolderInformationType" minOccurs="0"/>
    <xs:element name="EffectiveRights" type="t:EffectiveRightsType"
minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="BasePermissionType" abstract="true">
  <xs:sequence>
    <xs:element name="UserId" type="t:UserIdType"/>
    <xs:element name="CanCreateItems" type="xs:boolean" minOccurs="0"/>
    <xs:element name="CanCreateSubFolders" type="xs:boolean" minOccurs="0"/>
    <xs:element name="IsFolderOwner" type="xs:boolean" minOccurs="0"/>
    <xs:element name="IsFolderVisible" type="xs:boolean" minOccurs="0"/>
    <xs:element name="IsFolderContact" type="xs:boolean" minOccurs="0"/>
    <xs:element name="EditItems" type="t:PermissionActionType" minOccurs="0"/>
    <xs:element name="DeleteItems" type="t:PermissionActionType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="DeleteFolderFieldType">
  <xs:complexContent>
    <xs:extension base="t:FolderChangeDescriptionType">
      <xs:sequence/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="FolderChangeDescriptionType">
  <xs:complexContent>
    <xs:extension base="t:ChangeDescriptionType"/>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="FolderChangeType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="FolderId" type="t:FolderIdType"/>
      <xs:element name="DistinguishedFolderId"
type="t:DistinguishedFolderIdType"/>
    </xs:choice>
    <xs:element name="Updates"
type="t:NonEmptyArrayOfFolderChangeDescriptionsType"/>
  </xs:sequence>
</xs:complexType>

```

```

<xs:simpleType name="FolderClassType">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
<xs:complexType name="ManagedFolderInformationType">
  <xs:sequence>
    <xs:element name="CanDelete" type="xs:boolean" minOccurs="0"/>
    <xs:element name="CanRenameOrMove" type="xs:boolean" minOccurs="0"/>
    <xs:element name="MustDisplayComment" type="xs:boolean" minOccurs="0"/>
    <xs:element name="HasQuota" type="xs:boolean" minOccurs="0"/>
    <xs:element name="IsManagedFoldersRoot" type="xs:boolean" minOccurs="0"/>
    <xs:element name="ManagedFolderId" type="xs:string" minOccurs="0"/>
    <xs:element name="Comment" type="xs:string" minOccurs="0"/>
    <xs:element name="StorageQuota" type="xs:int" minOccurs="0"/>
    <xs:element name="FolderSize" type="xs:int" minOccurs="0"/>
    <xs:element name="HomePage" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="FolderType">
  <xs:complexContent>
    <xs:extension base="t:BaseFolderType">
      <xs:sequence>
        <xs:element name="PermissionSet" type="t:PermissionSetType"
minOccurs="0"/>
        <xs:element name="UnreadCount" type="xs:int" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="NonEmptyArrayOfBaseFolderIdsType">
  <xs:choice maxOccurs="unbounded">
    <xs:element name="FolderId" type="t:FolderIdType"/>
    <xs:element name="DistinguishedFolderId" type="t:DistinguishedFolderIdType"/>
  </xs:choice>
</xs:complexType>
<xs:complexType name="NonEmptyArrayOfFolderChangeDescriptionsType">
  <xs:choice maxOccurs="unbounded">
    <xs:element name="AppendToFolderField" type="t:AppendToFolderFieldType"/>
    <xs:element name="SetFolderField" type="t:SetFolderFieldType"/>
    <xs:element name="DeleteFolderField" type="t>DeleteFolderFieldType"/>
  </xs:choice>
</xs:complexType>
<xs:complexType name="NonEmptyArrayOfFolderChangesType">
  <xs:sequence>
    <xs:element name="FolderChange" type="t:FolderChangeType"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="NonEmptyArrayOfFolderNamesType">
  <xs:sequence>
    <xs:element name="FolderName" type="xs:string" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="NonEmptyArrayOfFoldersType">
  <xs:choice maxOccurs="unbounded">
    <xs:element name="Folder" type="t:FolderType"/>
    <xs:element name="CalendarFolder" type="t:CalendarFolderType"/>
    <xs:element name="ContactsFolder" type="t:ContactsFolderType"/>
    <xs:element name="SearchFolder" type="t:SearchFolderType"/>
    <xs:element name="TasksFolder" type="t:TasksFolderType"/>
  </xs:choice>
</xs:complexType>

```

```

        </xs:choice>
    </xs:complexType>
    <xs:simpleType name="PermissionActionType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="Owned"/>
            <xs:enumeration value="All"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="PermissionLevelType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="Owner"/>
            <xs:enumeration value="PublishingEditor"/>
            <xs:enumeration value="Editor"/>
            <xs:enumeration value="PublishingAuthor"/>
            <xs:enumeration value="Author"/>
            <xs:enumeration value="NoneditingAuthor"/>
            <xs:enumeration value="Reviewer"/>
            <xs:enumeration value="Contributor"/>
            <xs:enumeration value="Custom"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="PermissionReadAccessType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="FullDetails"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="PermissionSetType">
        <xs:sequence>
            <xs:element name="Permissions" type="t:ArrayOfPermissionsType"/>
            <xs:element name="UnknownEntries" type="t:ArrayOfUnknownEntriesType"
minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PermissionType">
        <xs:complexContent>
            <xs:extension base="t:BasePermissionType">
                <xs:sequence>
                    <xs:element name="ReadItems" type="t:PermissionReadAccessType"
minOccurs="0"/>
                    <xs:element name="PermissionLevel" type="t:PermissionLevelType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="SetFolderFieldType">
        <xs:complexContent>
            <xs:extension base="t:FolderChangeDescriptionType">
                <xs:choice>
                    <xs:element name="Folder" type="t:FolderType"/>
                    <xs:element name="CalendarFolder" type="t:CalendarFolderType"/>
                    <xs:element name="ContactsFolder" type="t:ContactsFolderType"/>
                    <xs:element name="SearchFolder" type="t:SearchFolderType"/>
                    <xs:element name="TasksFolder" type="t:TasksFolderType"/>
                </xs:choice>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

```

```

</xs:complexType>
<xs:complexType name="TargetFolderIdType">
  <xs:choice>
    <xs:element name="FolderId" type="t:FolderIdType"/>
    <xs:element name="DistinguishedFolderId" type="t:DistinguishedFolderIdType"/>
  </xs:choice>
</xs:complexType>
</xs:schema>

```

6.3 Messages Schema

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

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

Defining specification	File name
[MS-OXWSCDATA] , section 6.3	MS-OXWSCDATA -messages.xsd

```

<?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="Exchange2010" id="messages">
  <xs:import namespace="http://schemas.microsoft.com/exchange/services/2006/types"
    schemaLocation="MS-OXWSFOLD-TYPES.xsd"/>
  <xs:include schemaLocation="MS-OXWSCDATA-messages.xsd"/>
  <xs:complexType name="BaseMoveCopyFolderType">
    <xs:complexContent>
      <xs:extension base="m:BaseRequestType">
        <xs:sequence>
          <xs:element name="ToFolderId" type="t:TargetFolderIdType"/>
          <xs:element name="FolderIds"
            type="t:NonEmptyArrayOfBaseFolderIdsType"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="CopyFolder" type="m:CopyFolderType"/>
  <xs:complexType name="CopyFolderType">
    <xs:complexContent>
      <xs:extension base="m:BaseMoveCopyFolderType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="CopyFolderResponse" type="m:CopyFolderResponseType"/>
  <xs:complexType name="CopyFolderResponseType">
    <xs:complexContent>
      <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="CreateFolderResponseType">

```

```

        <xs:complexContent>
            <xs:extension base="m:BaseResponseMessageType"/>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CreateFolderResponse" type="m:CreateFolderResponseType"/>
    <xs:complexType name="CreateFolderType">
        <xs:complexContent>
            <xs:extension base="m:BaseRequestType">
                <xs:sequence>
                    <xs:element name="ParentFolderId" type="t:TargetFolderIdType"/>
                    <xs:element name="Folders" type="t:NonEmptyArrayOfFoldersType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CreateFolder" type="m:CreateFolderType"/>
    <xs:complexType name="CreateManagedFolderRequestType">
        <xs:complexContent>
            <xs:extension base="m:BaseRequestType">
                <xs:sequence>
                    <xs:element name="FolderNames"
type="t:NonEmptyArrayOfFolderNamesType"/>
                    <xs:element name="Mailbox" type="t:EmailAddressType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CreateManagedFolder" type="m:CreateManagedFolderRequestType"/>
    <xs:complexType name="CreateManagedFolderResponseType">
        <xs:complexContent>
            <xs:extension base="m:BaseResponseMessageType"/>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="CreateManagedFolderResponse"
type="m:CreateManagedFolderResponseType"/>
    <xs:complexType name="DeleteFolderType">
        <xs:complexContent>
            <xs:extension base="m:BaseRequestType">
                <xs:sequence>
                    <xs:element name="FolderIds"
type="t:NonEmptyArrayOfBaseFolderIdsType"/>
                </xs:sequence>
                <xs:attribute name="DeleteType" type="t:DisposalType" use="required"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="DeleteFolder" type="m:DeleteFolderType"/>
    <xs:complexType name="DeleteFolderResponseType">
        <xs:complexContent>
            <xs:extension base="m:BaseResponseMessageType"/>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="DeleteFolderResponse" type="m:DeleteFolderResponseType"/>
    <xs:complexType name="FolderInfoResponseType">
        <xs:complexContent>
            <xs:extension base="m:ResponseType">
                <xs:sequence>
                    <xs:element name="Folders" type="t:ArrayOfFoldersType"
minOccurs="0"/>

```



```

        </xs:sequence>
    </xs:extension>
</xs:complexType>
</xs:complexType>
<xs:complexType name="GetFolderType">
    <xs:complexContent>
        <xs:extension base="m:BaseRequestType">
            <xs:sequence>
                <xs:element name="FolderShape" type="t:FolderResponseShapeType"/>
                <xs:element name="FolderIds"
type="t:NonEmptyArrayOfBaseFolderIdsType"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:element name="GetFolder" type="m:GetFolderType"/>
<xs:complexType name="GetFolderResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
<xs:element name="GetFolderResponse" type="m:GetFolderResponseType"/>
<xs:complexType name="MoveFolderResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
<xs:element name="MoveFolderResponse" type="m:MoveFolderResponseType"/>
<xs:complexType name="MoveFolderType">
    <xs:complexContent>
        <xs:extension base="m:BaseMoveCopyFolderType"/>
    </xs:complexContent>
</xs:complexType>
<xs:element name="MoveFolder" type="m:MoveFolderType"/>
<xs:complexType name="UpdateFolderType">
    <xs:complexContent>
        <xs:extension base="m:BaseRequestType">
            <xs:sequence>
                <xs:element name="FolderChanges"
type="t:NonEmptyArrayOfFolderChangesType"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:element name="UpdateFolder" type="m:UpdateFolderType"/>
<xs:complexType name="UpdateFolderResponseType">
    <xs:complexContent>
        <xs:extension base="m:BaseResponseMessageType"/>
    </xs:complexContent>
</xs:complexType>
<xs:element name="UpdateFolderResponse" type="m:UpdateFolderResponseType"/>
</xs:schema>

```

7 Appendix B: Product Behavior

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

- Microsoft® Exchange Server 2010

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

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

8 Change Tracking

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
1.3 Overview	Updated the section title.	N	Content updated for template compliance.
3.1.4.1.2.2 m:CopyFolderType Complex Type	Added information about the base type.	Y	New content added.
3.1.4.6.2.2 m:MoveFolderType Complex Type	Added information about the base type.	N	New content added.

9 Index

A

[Applicability](#) 9

C

[Capability negotiation](#) 9

[Change tracking](#) 75

Client

[abstract data model](#) 53

[initialization](#) 53

[local events](#) 54

[message processing](#) 54

[overview](#) 53

[sequencing rules](#) 54

[timer events](#) 54

[timers](#) 53

F

[Full WSDL](#) 62

G

[Glossary](#) 7

I

[Introduction](#) 7

M

Messages

[overview](#) 10

[syntax](#) 10

[transport](#) 10

O

[Overview](#) 8

P

[Preconditions](#) 9

[Prerequisites](#) 9

[Product behavior](#) 74

R

[Relationship to other protocols](#) 8

S

Security

[implementer considerations](#) 61

[overview](#) 61

[parameter index](#) 61

Server

[abstract data model](#) 26

[initialization](#) 26

[local events](#) 53

[message processing](#) 26

[overview](#) 26

[sequencing rules](#) 26

[timer events](#) 53

[timers](#) 26

[Standards assignments](#) 9

T

[Tracking changes](#) 75

V

[Vendor-extensible fields](#) 9

[Versioning](#) 9