

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Revised and edited the technical content
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Minor	Updated the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Minor	Clarified the meaning of the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.5	Minor	Clarified the meaning of the technical content.

Table of Contents

1	Introduction	6
1.1	Glossary	6
1.2	References.....	6
1.2.1	Normative References.....	6
1.2.2	Informative References	7
1.3	Protocol Overview (Synopsis)	7
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	Messages	10
2.2.3	Elements.....	10
2.2.4	Complex Types	10
2.2.4.1	ArrayOfString.....	11
2.2.4.2	ArrayOfRows.....	11
2.2.4.3	SOAPFaultDetails.....	11
2.2.5	Simple Types	12
2.2.6	Attributes.....	12
2.2.7	Groups.....	12
2.2.8	Attribute Groups	12
3	Protocol Details.....	13
3.1	ImagingSoap Server Details.....	13
3.1.1	Abstract Data Model	13
3.1.2	Timers	13
3.1.3	Initialization	13
3.1.4	Message Processing Events and Sequencing Rules.....	13
3.1.4.1	CheckSubwebAndList	14
3.1.4.1.1	Messages	14
3.1.4.1.1.1	CheckSubwebAndListSoapIn	14
3.1.4.1.1.2	CheckSubwebAndListSoapOut	14
3.1.4.1.2	Elements.....	15
3.1.4.1.2.1	CheckSubwebAndList	15
3.1.4.1.2.2	CheckSubwebAndListResponse.....	15
3.1.4.2	CreateNewFolder	16
3.1.4.2.1	Messages	16
3.1.4.2.1.1	CreateNewFolderSoapIn	16
3.1.4.2.1.2	CreateNewFolderSoapOut	17
3.1.4.2.2	Elements.....	17
3.1.4.2.2.1	CreateNewFolder	17
3.1.4.2.2.2	CreateNewFolderResponse.....	17
3.1.4.3	Delete.....	18
3.1.4.3.1	Messages	18

3.1.4.3.1.1	DeleteSoapIn	18
3.1.4.3.1.2	DeleteSoapOut	18
3.1.4.3.2	Elements	19
3.1.4.3.2.1	Delete	19
3.1.4.3.2.2	DeleteResponse	19
3.1.4.3.3	Complex Types	20
3.1.4.3.3.1	ArrayOfDeleteResults	20
3.1.4.4	Download	20
3.1.4.4.1	Messages	21
3.1.4.4.1.1	DownloadSoapIn	21
3.1.4.4.1.2	DownloadSoapOut	21
3.1.4.4.2	Elements	21
3.1.4.4.2.1	Download	21
3.1.4.4.2.2	DownloadResponse	22
3.1.4.4.3	Complex Types	22
3.1.4.4.3.1	ArrayOfFiles	22
3.1.4.5	Edit	23
3.1.4.5.1	Messages	23
3.1.4.5.1.1	EditSoapIn	23
3.1.4.5.1.2	EditSoapOut	23
3.1.4.5.2	Elements	24
3.1.4.5.2.1	Edit	24
3.1.4.5.2.2	EditResponse	24
3.1.4.6	GetItemsByIds	24
3.1.4.6.1	Messages	25
3.1.4.6.1.1	GetItemsByIdsSoapIn	25
3.1.4.6.1.2	GetItemsByIdsSoapOut	25
3.1.4.6.2	Elements	25
3.1.4.6.2.1	GetItemsByIds	25
3.1.4.6.2.2	GetItemsByIdsResponse	26
3.1.4.6.3	Complex Types	26
3.1.4.6.3.1	ArrayOfUnsignedInt	26
3.1.4.7	GetItemsXMLData	26
3.1.4.7.1	Messages	27
3.1.4.7.1.1	GetItemsXMLDataSoapIn	27
3.1.4.7.1.2	GetItemsXMLDataSoapOut	27
3.1.4.7.2	Elements	27
3.1.4.7.2.1	GetItemsXMLData	27
3.1.4.7.2.2	GetItemsXMLDataResponse	28
3.1.4.7.3	Complex Types	28
3.1.4.7.3.1	ArrayOfItems	28
3.1.4.8	GetListItems	29
3.1.4.8.1	Messages	30
3.1.4.8.1.1	GetListItemsSoapIn	30
3.1.4.8.1.2	GetListItemsSoapOut	30
3.1.4.8.2	Elements	30
3.1.4.8.2.1	GetListItems	30
3.1.4.8.2.2	GetListItemsResponse	31
3.1.4.9	ListPictureLibrary	31
3.1.4.9.1	Messages	32
3.1.4.9.1.1	ListPictureLibrarySoapIn	32
3.1.4.9.1.2	ListPictureLibrarySoapOut	32
3.1.4.9.2	Elements	32

3.1.4.9.2.1	ListPictureLibrary	32
3.1.4.9.2.2	ListPictureLibraryResponse	32
3.1.4.9.3	Complex Types	33
3.1.4.9.3.1	ArrayOfLibraries	33
3.1.4.10	Rename	33
3.1.4.10.1	Messages	34
3.1.4.10.1.1	RenameSoapIn	34
3.1.4.10.1.2	RenameSoapOut	34
3.1.4.10.2	Elements	34
3.1.4.10.2.1	Rename	34
3.1.4.10.2.2	RenameResponse	35
3.1.4.10.3	Complex Types	35
3.1.4.10.3.1	ArrayOfRenameFiles	35
3.1.4.10.3.2	ArrayOfRenameResults	35
3.1.4.11	Upload	36
3.1.4.11.1	Messages	37
3.1.4.11.1.1	UploadSoapIn	37
3.1.4.11.1.2	UploadSoapOut	37
3.1.4.11.2	Elements	37
3.1.4.11.2.1	Upload	37
3.1.4.11.2.2	UploadResponse	38
3.1.5	Timer Events	38
3.1.6	Other Local Events	38
4	Protocol Examples	39
5	Security	43
5.1	Security Considerations for Implementers	43
5.2	Index of Security Parameters	43
6	Appendix A: Full WSDL	44
7	Appendix B: Product Behavior	57
8	Change Tracking	58
9	Index	60

1 Introduction

This document specifies the Imaging Service Protocol. This protocol specifies a set of client-server interactions that allows a client to retrieve, upload, and organize images on a server.

1.1 Glossary

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

Coordinated Universal Time (UTC)
GUID
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)

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

file
folder
item
list
list item
picture library
Simple Object Access Protocol (SOAP)
site
SOAP action
SOAP body
SOAP fault
subsite
Uniform Resource Locator (URL)
URL encode
Web Services Description Language (WSDL)
WSDL message
WSDL operation
XML namespace
XML namespace prefix
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-LISTSWS] Microsoft Corporation, "[Lists Web Service Protocol Specification](#)"

[MS-PRSTFR] Microsoft Corporation, "[ADO XML Persistence Format Protocol Specification](#)"

[MS-WSSFO2] Microsoft Corporation, "[Windows SharePoint Services \(WSS\): File Operations Database Communications Version 2 Protocol Specification](#)".

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

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006, <http://www.ietf.org/rfc/rfc4648.txt>

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

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624>

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part2-20030624>

[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., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208/>

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

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

1.2.2 Informative References

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

[MS-OFCGLOS] Microsoft Corporation, "[Microsoft Office Master Glossary](#)".

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, <http://www.ietf.org/rfc/rfc2818.txt>

1.3 Protocol Overview (Synopsis)

This protocol is a **SOAP**-based protocol that uses HTTP/S as its transport. Protocol clients can perform the following actions:

- Attempt to have a specified **URL** resolved as a **subsite**, **list (1)**, and **folder** for a **picture library**.
- Create a new folder in a picture library.
- Delete **list items** from a picture library.

- Download list items from a picture library.
- Provide identifiers and have the corresponding list items in a picture library returned.
- Provide names and have the corresponding metadata of the list items in a picture library returned.
- Obtain all list items in a specified folder in a picture library.
- List all picture library items on the **site (2)**.
- Rename list items in a picture library.
- Upload an image to a folder in a picture library.

The following diagram shows the communication between the protocol client and the protocol server:

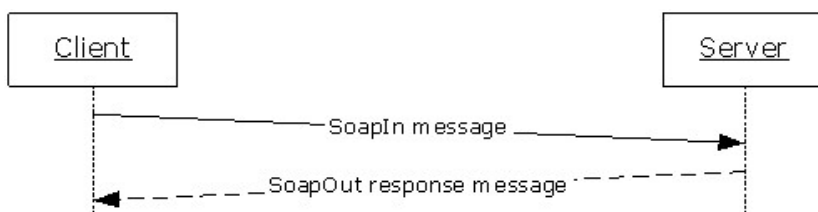


Figure 1: Sample WSDL operation sequence

The protocol client sends a request **WSDL message** to the protocol server.

The protocol server replies with a response WSDL message to the protocol client. If the **WSDL operation** succeeds, the response WSDL message of the WSDL operation is sent. If the WSDL operation fails, a **SOAP fault** WSDL message is sent with an error code.

1.4 Relationship to Other Protocols

This protocol uses the SOAP message protocol for formatting request and response messages, as described in [\[SOAP1.1\]](#), [\[SOAP1.2/1\]](#) and [\[SOAP1.2/2\]](#). It transmits those messages by using **HTTP**, as described in [\[RFC2616\]](#), or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [\[RFC2818\]](#).

The following diagram shows the underlying messaging and transport stack used by the protocol:

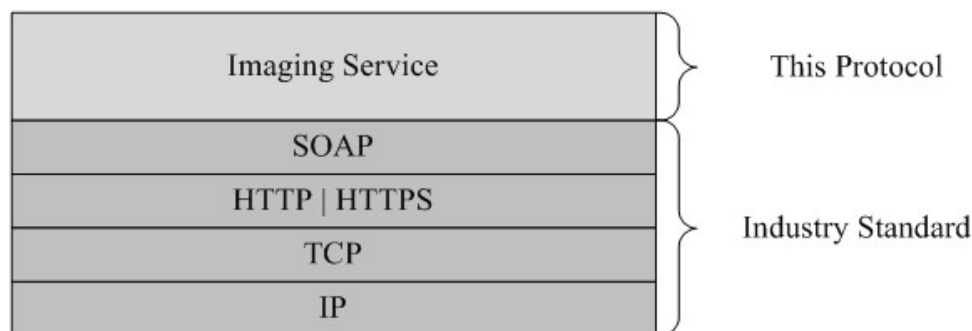


Figure 2: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a site (2) that is identified by a URL that is known by protocol clients. The protocol server endpoint is formed by appending "_vti_bin/imaging.asmx" to the URL of the site (2), for example http://www.contoso.com/Repository/_vti_bin/Imaging.asmx.

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

None.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses multiple transports with SOAP as specified in section [2.1](#).

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers **MUST** support SOAP over HTTP. Protocol servers **SHOULD** additionally support SOAP over HTTPS for securing communication with clients.

Protocol messages **MUST** be formatted as specified either in [\[SOAP1.1\]](#), section 4 or in [\[SOAP1.2/1\]](#), section 5. Protocol server faults **MUST** be returned using either HTTP Status Codes as specified in [\[RFC2616\]](#), section 10 or SOAP faults, as specified in [\[SOAP1.1\]](#), section 4.4 or [\[SOAP1.2/1\]](#), section 5.4.

2.2 Common Message Syntax

This section contains common structures used by this protocol. The syntax of the structures uses XML Schema as defined in [\[XMLSCHEMA1\]](#) and [\[XMLSCHEMA2\]](#), and **WSDL** as defined in [\[WSDL\]](#).

2.2.1 Namespaces

This specification specifies and references various **XML namespaces**, using the mechanisms specified in [\[XMLNS\]](#). Although this specification associates a specific **XML namespace prefix** with 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/sharepoint/soap/ois/	
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
(none)	http://schemas.microsoft.com/sharepoint/soap/ois/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
z	#RowsetSchema	[MS-PRSTFR]

2.2.2 Messages

None.

2.2.3 Elements

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

2.2.4 Complex Types

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

Complex Type	Description
ArrayOfString	An array of strings.
ArrayOfRows	An array of z:row objects.
SOAPFaultDetails	The details of a SOAP fault.

2.2.4.1 ArrayOfString

An array of strings.

```
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element name="string" type="s:string" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

string: MUST be **URL-encoded**. MUST be non-empty and less than 256 characters in length.

2.2.4.2 ArrayOfRows

An array of **z:row** elements. **z** is equal to #RowsetSchema as specified in [\[MS-PRSTFR\]](#). Each **z:row** element describes a single list item. The names of the attributes of this element correspond to the names of fields in the list (1). (For more details, see [\[MS-WSSFO2\]](#).) To get the schema of the list (1), protocol clients can make a **GetList** call as specified in [\[MS-LISTSWS\]](#) section 3.1.4.15.

```
<s:complexType name="ArrayOfRows" mixed="true">
  <s:sequence>
    <s:any minOccurs="0" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

2.2.4.3 SOAPFaultDetails

The details of a SOAP fault.

```
<s:schema xmlns:s="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://schemas.microsoft.com/sharepoint/soap">
  <s:complexType name="SOAPFaultDetails">
    <s:sequence>
      <s:element name="errorstring" type="s:string"/>
      <s:element name="errorcode" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:schema>
```

errorstring: A human-readable text explaining the application-level fault.

errorcode: The hexadecimal representation of a 4-byte result code.

Errorcode	Condition
0x00000001	ListNotFound: The requested list (1) was not found.
0x00000002	IsNotLibrary: The requested list (1), although found, is not a picture library.
0x00000004	FolderNotFound: The requested folder was not found.
0x00000005	InvalidArgument: One or more arguments are not valid, specified as follows.
0x00000006	FileExists: The file (1) already exists, and the user does not specify the overwrite option. IllegalFileName: The file name contains illegal character(s), specified as follows.
0x81020067	EmptyString: The file name is an empty string.

For the folder name in the request WSDL message, the server MUST respond with an **InvalidArgument** error code in the following conditions:

- The folder name contains illegal characters, specified as: \ : * ? " < > | # { } % and \t.
- The folder name contains a parent path directive—that is, "..".
- The relative path contains a folder named "forms".
- The relative path contains a folder named "_t" or "_w".

For the file name in the request WSDL message, the server MUST respond with an **InvalidArgument** error code if the file name contains any of the following characters:

/, \, \t

For the file name in the request WSDL message, the server MUST respond with an **IllegalFileName** error code if the file name contains any of the following characters:

:, *, ?, ", <, >, |, #, {, }, %

2.2.5 Simple Types

This specification does not define any common XML Schema simple type definitions.

2.2.6 Attributes

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

2.2.7 Groups

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

2.2.8 Attribute Groups

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

3 Protocol Details

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

Except where specified, protocol clients SHOULD interpret HTTP Status Codes returned by the protocol server as specified in [\[RFC2616\]](#), section 10.

This protocol enables protocol servers to notify protocol clients of application-level faults using SOAP faults. This protocol enables protocol servers to provide additional details for SOAP faults by including a detail element as specified in [\[SOAP1.1\]](#), section 4.4 that conforms to the **XML schema** of the **SOAPFaultDetails** complex type specified in **SOAPFaultDetails** (section [2.2.4.3](#)). Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

This protocol enables protocol servers to perform implementation-specific authorization checks and to notify protocol clients of authorization faults either using HTTP Status Codes or using SOAP faults as specified previously in this section.

3.1 ImagingSoap Server Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

Operation	Description
CheckSubwebAndList	A WSDL operation that checks the specified URL and attempts to resolve it as a subsite, list (1), and folder.
CreateNewFolder	A WSDL operation that creates a new folder in the specified list (1) and folder.
Delete	A WSDL operation that removes the specified list items from the specified list (1) and folder.
Download	A WSDL operation that downloads the specified list items from the specified list (1) and folder.
Edit	A WSDL operation that edits an image in the specified list (1) and folder. This WSDL operation is reserved.
GetItemsByIds	A WSDL operation that returns list items with the specified identifiers in the specified list (1).

Operation	Description
GetItemsXMLData	A WSDL operation that returns metadata of specified list items in the specified list (1) and folder.
GetListItems	A WSDL operation that returns list items in the specified list (1) and folder.
ListPictureLibrary	A WSDL operation that returns all picture libraries on the current site (2).
Rename	A WSDL operation that renames items in a list (1) and folder.
Upload	A WSDL operation that uploads a file to the specified list (1) and folder.

3.1.4.1 CheckSubwebAndList

This WSDL operation checks the specified URL and attempts to resolve it as a subsite, list (1), and folder.

```
<wsdl:operation name="CheckSubwebAndList">
  <wsdl:input message="CheckSubwebAndListSoapIn" />
  <wsdl:output message="CheckSubwebAndListSoapOut" />
</wsdl:operation>
```

The client sends a **CheckSubwebAndListSoapIn** request WSDL message, and the server responds with a **CheckSubwebAndListSoapOut** response WSDL message, as follows:

- If the list (1) contained in the **strUrl**, although found, is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- Otherwise, the server MUST send a **CheckSubwebAndListSoapOut** response WSDL message.

3.1.4.1.1 Messages

3.1.4.1.1.1 CheckSubwebAndListSoapIn

This WSDL message contains information required by the **CheckSubwebAndList** WSDL operation. This WSDL message is a request.

The **SOAP action** value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/CheckSubwebAndList
```

The **SOAP body** contains a CheckSubwebAndList element.

3.1.4.1.1.2 CheckSubwebAndListSoapOut

This WSDL message contains results returned by the **CheckSubwebAndList** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/CheckSubwebAndList
```

The SOAP body contains a **CheckSubwebAndListResponse** element.

3.1.4.1.2 Elements

3.1.4.1.2.1 CheckSubwebAndList

Input data for a **CheckSubwebAndList** WSDL operation.

```
<s:element name="CheckSubwebAndList">
  <s:complexType>
    <s:sequence>
      <s:element name="strUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strUrl: The URL to check. MUST contain a valid URL. MUST be URL-encoded.

3.1.4.1.2.2 CheckSubwebAndListResponse

Result data for a **CheckSubwebAndList** WSDL operation.

```
<s:element name="CheckSubwebAndListResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="CheckSubwebAndListResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="result">
              <s:complexType>
                <s:attribute name="url" type="s:string"/>
                <s:attribute name="subweb" type="s:string"/>
                <s:attribute name="list" type="s:string"/>
                <s:attribute name="listGuid" type="s:string"/>
                <s:attribute name="folder" type="s:string"/>
                <s:attribute name="rest" type="s:string"/>
                <s:attribute name="found" type="s:boolean" use="optional" default="true"/>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

CheckSubwebAndListResult: The element containing the result of the **CheckSubwebAndList** WSDL operation.

CheckSubwebAndListResult.result: The result of the **CheckSubwebAndList** WSDL operation.

CheckSubwebAndListResult.result.found: True if the subsite was found; otherwise, false.

CheckSubwebAndListResult.result.url: The original URL in the request WSDL message.

CheckSubwebAndListResult.result.subweb: The URL of the subsite.

CheckSubwebAndListResult.result.list: The name of the list (1) in the requested URL, if the requested URL is part of a list (1).

CheckSubwebAndListResult.result.listGuid: The **GUID** of the list (1) in the requested URL, if the requested URL is part of a list (1).

CheckSubwebAndListResult.result.folder: The name of the folder in the requested URL, if the requested URL includes a folder.

CheckSubwebAndListResult.result.rest: The remaining path of the requested URL after the **subweb**, list (1), or folder elements. The default is null if there is no remaining path after the **subweb**, list (1), or folder elements.

3.1.4.2 CreateNewFolder

This WSDL operation creates a new folder in the specified list (1) and folder.

```
<wsdl:operation name="CreateNewFolder">
  <wsdl:input message="CreateNewFolderSoapIn" />
  <wsdl:output message="CreateNewFolderSoapOut" />
</wsdl:operation>
```

The client sends a **CreateNewFolderSoapIn** request WSDL message, and the server responds with a **CreateNewFolderSoapOut** response WSDL message, as follows:

- If the **strListName** does not exist or is empty, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strParentFolder** is not a legal folder path as specified in **SOAPFaultDetails**, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strParentFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- Otherwise, the server MUST send a **CreateNewFolderSoapOut** response and MUST create the new folder with an unused name, either "New folder" or "New folder (*n*)" if a folder with that name already exists, where *n* is a number greater than 0.

3.1.4.2.1 Messages

3.1.4.2.1.1 CreateNewFolderSoapIn

This WSDL message contains information required by the **CreateNewFolder** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/CreateNewFolder
```

The SOAP body contains a **CreateNewFolder** element.

3.1.4.2.1.2 CreateNewFolderSoapOut

This WSDL message contains results returned by the **CreateNewFolder** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/CreateNewFolder
```

The SOAP body contains a **CreateNewFolderResponse** element.

3.1.4.2.2 Elements

3.1.4.2.2.1 CreateNewFolder

Input data for a **CreateNewFolder** WSDL operation.

```
<s:element name="CreateNewFolder">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strParentFolder" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST be non-empty.

strParentFolder: The relative path from the root of the list (1) to the parent folder. MUST be URL-encoded.

3.1.4.2.2.2 CreateNewFolderResponse

Result data for a **CreateNewFolder** WSDL operation.

```
<s:element name="CreateNewFolderResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="CreateNewFolderResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="NewFolder">
              <s:complexType>
                <s:attribute name="title" type="s:string"/>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

CreateNewFolderResult: The result of the **CreateNewFolder** WSDL operation.

CreateNewFolderResult.NewFolder: The new folder that is created by the **CreateNewFolder** WSDL operation.

CreateNewFolderResult.NewFolder.title: The title of the new folder.

3.1.4.3 Delete

This WSDL operation removes the specified list items from the specified list (1) and folder.

```
<wsdl:operation name="Delete">
  <wsdl:input message="DeleteSoapIn" />
  <wsdl:output message="DeleteSoapOut" />
</wsdl:operation>
```

The client sends a **DeleteSoapIn** request WSDL message, and the server responds with a **DeleteSoapOut** response WSDL message, as follows:

- If **strListName** or **itemFileNames** is empty, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- If **itemFileNames** contains any illegal character(s) as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **IllegalFileName** SOAP fault.
- Otherwise, the server MUST send a **DeleteSoapOut** response WSDL message with the **deleted** flag set to specify WSDL operation status. The server MUST delete the specified list items.

3.1.4.3.1 Messages

3.1.4.3.1.1 DeleteSoapIn

This WSDL message contains information required by the Delete WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Delete
```

The SOAP body contains a **Delete** element.

3.1.4.3.1.2 DeleteSoapOut

This WSDL message contains results returned by the **Delete** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Delete
```

The SOAP body contains a **DeleteResponse** element.

3.1.4.3.2 Elements

3.1.4.3.2.1 Delete

Input data for a **Delete** WSDL operation.

```
<s:element name="Delete">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
      <s:element name="itemFileNames" type="tns:ArrayOfString"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) in the current site (2). MUST be URL-encoded. MUST NOT be empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded. An empty value indicates the root of the list (1).

itemFileNames: An array of the names of list items to be deleted from the list (1). MUST NOT be empty.

3.1.4.3.2.2 DeleteResponse

Result data for a **Delete** WSDL operation.

```
<s:element name="DeleteResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="DeleteResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfDeleteResults"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

DeleteResult: The container element of **DeleteResult.results**.

DeleteResult.results: The result of the **Delete** WSDL operation.

3.1.4.3.3 Complex Types

3.1.4.3.3.1 ArrayOfDeleteResults

Array of **Delete** WSDL operation results.

```
<s:complexType name="ArrayOfDeleteResults">
  <s:sequence>
    <s:element name="result" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="deleted" type="s:boolean"/>
        <s:attribute name="name" type="s:string"/>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>
```

result: Contains the result of the WSDL operation.

result.deleted: True if the list item was successfully deleted; otherwise, false.

result.name: The name of the list item that was deleted.

3.1.4.4 Download

This WSDL operation downloads the specified list items from the specified list (1) and folder.

```
<wsdl:operation name="Download">
  <wsdl:input message="DownloadSoapIn" />
  <wsdl:output message="DownloadSoapOut" />
</wsdl:operation>
```

The client sends a **DownloadSoapIn** request WSDL message, and the server responds with a **DownloadSoapOut** response WSDL message, as follows:

- If **strListName** or **itemFileNames** is empty or the **type** is not an integer between 0 to 2 inclusive, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- If **itemFileNames** contains any illegal character(s) as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **IllegalFileName** SOAP fault.
- Otherwise, the server MUST send a **DownloadSoapOut** response with the **found** flag set to specify WSDL operation status.

3.1.4.4.1 Messages

3.1.4.4.1.1 DownloadSoapIn

This WSDL message contains information required by the **Download** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Download
```

The SOAP body contains a **Download** element.

3.1.4.4.1.2 DownloadSoapOut

This WSDL message contains results returned by the Download WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Download
```

The SOAP body contains a **DownloadResponse** element.

3.1.4.4.2 Elements

3.1.4.4.2.1 Download

Input data for a **Download** WSDL operation.

```
<s:element name="Download">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
      <s:element name="itemFileNames" type="tns:ArrayOfString"/>
      <s:element name="type" type="s:unsignedInt"/>
      <s:element name="fetchOriginalIfNotAvailable" type="s:boolean"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST NOT be empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded. An empty value indicates the root of the list (1).

itemFileNames: An array of the names of list items to download. MUST NOT be empty.

type: The version of the files to download. The following table shows valid values.

Value	Version
0	The original image in the original file format when it was uploaded. The width and height of this image is the same as the original image uploaded to the server.
1	The thumbnail of the image in JPEG format. The width and height of this image are determined during the JPEG compression phase and are not fixed numbers.
2	The image optimized for use on the Web in JPEG format. The width and height of this image are determined during the JPEG compression phase and are not fixed numbers.

The image width and height for values 1 and 2 are calculated depending on the width and height of the original image during JPEG compression.

fFetchOriginalIfNotAvailable: A value that MUST be TRUE or FALSE. If this parameter is set to TRUE and the requested version is not available for any reason, the original version MUST be returned, and the **File** element will have an **originalDownloaded** attribute set to TRUE. If this parameter is set to FALSE and the requested version is not available, an error code of 0x81070211 will be returned, indicating that the file could not be opened.

3.1.4.4.2 DownloadResponse

Result data for a **Download** WSDL operation.

```
<s:element name="DownloadResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="DownloadResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="Files" type="tns:ArrayOfFiles"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

DownloadResult.Files: The result of the **Download** WSDL operation.

3.1.4.4.3 Complex Types

3.1.4.4.3.1 ArrayOfFiles

Array of file objects. The content of the file MUST be encoded with Base64, as specified in [\[RFC4648\]](#).

```
<s:complexType name="ArrayOfFiles">
  <s:sequence>
    <s:element name="File" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
        <s:simpleContent>
          <s:extension base="s:base64Binary">
            <s:attribute name="name" type="s:string"/>
            <s:attribute name="lastmodified" type="s:dateTime"/>
          </s:extension>
        </s:simpleContent>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>
```

```

        <s:attribute name="found" type="s:boolean" use="optional" default="true"/>
      </s:extension>
    </s:simpleContent>
  </s:complexType>
</s:element>
</s:sequence>
</s:complexType>

```

File: Contains information about a file.

File.name: The name of the file. MUST NOT be the name of a folder.

File.found: True if the list item is found; otherwise, false.

File.lastmodified: The time stamp of the last modification. MUST be in **UTC** format.

3.1.4.5 Edit

This WSDL operation edits an image in the specified list (1) and folder. This WSDL operation is reserved for future use.

```

<wsdl:operation name="Edit">
  <wsdl:input message="EditSoapIn" />
  <wsdl:output message="EditSoapOut" />
</wsdl:operation>

```

The client sends an **EditSoapIn** request WSDL message, and the server MUST respond with an **EditSoapOut** response WSDL message.

3.1.4.5.1 Messages

3.1.4.5.1.1 EditSoapIn

This WSDL message contains information required by the **Edit** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Edit
```

The SOAP body contains an **Edit** element.

3.1.4.5.1.2 EditSoapOut

This WSDL message contains results returned by the **Edit** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Edit
```

The SOAP body contains an **EditResponse** element.

3.1.4.5.2 Elements

3.1.4.5.2.1 Edit

Input data for an Edit WSDL operation.

```
<s:element name="Edit">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string" minOccurs="0"/>
      <s:element name="strFolder" type="s:string" minOccurs="0"/>
      <s:element name="itemFileName" type="s:string" minOccurs="0"/>
      <s:element name="recipe" minOccurs="0">
        <s:complexType>
          <s:sequence>
            <s:any/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2).

strFolder: The relative path from the root of the list (1) to the target folder.

itemFileName: The names of list items to edit.

recipe: The collection of edit commands.

3.1.4.5.2.2 EditResponse

Result data for an **Edit** WSDL operation.

```
<s:element name="EditResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="EditResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="notImplemented"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

notImplemented: This element **MUST** be set to empty when sent by the server, and **MUST** be ignored when received by the client.

3.1.4.6 GetItemsByIds

This WSDL operation returns list items with the specified identifier in the specified list (1).


```
<wsdl:operation name="GetItemsByIds">
  <wsdl:input message="GetItemsByIdsSoapIn" />
  <wsdl:output message="GetItemsByIdsSoapOut" />
</wsdl:operation>
```

The client sends a **GetItemsByIdsSoapIn** request WSDL message, and the server responds with a **GetItemsByIdsSoapOut** response WSDL message, as follows:

- If **strListName** or **ids** is empty, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- Otherwise, the server MUST send a **GetItemsByIdsSoapOut** response. If the specified identifiers are not found in the server, the server MUST ignore them and return the rest of the valid list items.

3.1.4.6.1 Messages

3.1.4.6.1.1 GetItemsByIdsSoapIn

This WSDL message contains information required by the **GetItemsByIds** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsByIds
```

The SOAP body contains a **GetItemsByIds** element.

3.1.4.6.1.2 GetItemsByIdsSoapOut

This WSDL message contains results returned by the **GetItemsByIds** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsByIds
```

The SOAP body contains a **GetItemsByIdsResponse** element.

3.1.4.6.2 Elements

3.1.4.6.2.1 GetItemsByIds

Input data for a **GetItemsByIds** WSDL operation.

```
<s:element name="GetItemsByIds">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="ids" type="tns:ArrayOfUnsignedInt"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

```

    </s:sequence>
  </s:complexType>
</s:element>

```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST NOT be empty.

ids: An array of the identifiers of the list items. MUST NOT be empty.

3.1.4.6.2.2 GetItemsByIdsResponse

Result data for a **GetItemsByIds** WSDL operation.

```

<s:element name="GetItemsByIdsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetItemsByIdsResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfRows"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>

```

GetItemsByIdsResult: The container element of the **GetItemsByIdsResult.results** element.

GetItemsByIdsResult.results: The array of results from the **GetItemsByIds** WSDL operation.

3.1.4.6.3 Complex Types

3.1.4.6.3.1 ArrayOfUnsignedInt

Array of unsigned integers.

```

<s:complexType name="ArrayOfUnsignedInt">
  <s:sequence>
    <s:element name="unsignedInt" type="s:unsignedInt" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>

```

unsignedInt: This element MUST be present.

3.1.4.7 GetItemsXMLData

This WSDL operation returns metadata of specified list items in the specified list (1) and folder.

```

<wsdl:operation name="GetItemsXMLData">
  <wsdl:input message="GetItemsXMLDataSoapIn" />
  <wsdl:output message="GetItemsXMLDataSoapOut" />
</wsdl:operation>

```

The client sends a **GetItemsXMLDataSoapIn** request WSDL message, and the server responds with a **GetItemsXMLDataSoapOut** response WSDL message, as follows:

- If **strListName** or **itemFileNames** is empty, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- If **itemFileNames** contains any illegal character(s) as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **IllegalFileName** SOAP fault.
- Otherwise, the server MUST send a **GetItemsXMLDataSoapOut** response.

3.1.4.7.1 Messages

3.1.4.7.1.1 GetItemsXMLDataSoapIn

This WSDL message contains information required by the **GetItemsXMLData** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsXMLData
```

The SOAP body contains a **GetItemsXMLData** element.

3.1.4.7.1.2 GetItemsXMLDataSoapOut

This WSDL message contains results returned by the **GetItemsXMLData** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsXMLData
```

The SOAP body contains a **GetItemsXMLDataResponse** element.

3.1.4.7.2 Elements

3.1.4.7.2.1 GetItemsXMLData

Input data for a **GetItemsXMLData** WSDL operation.

```
<s:element name="GetItemsXMLData">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

```

        <s:element name="strFolder" type="s:string"/>
        <s:element name="itemFileNames" type="tns:ArrayOfString"/>
    </s:sequence>
</s:complexType>
</s:element>

```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST be non-empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded. This specifies the root of the library if it is left empty.

itemFileNames: An array of the names of the list items queried for XML data.

3.1.4.7.2.2 GetItemsXMLDataResponse

Result data for a **GetItemsXMLData** WSDL operation.

```

<s:element name="GetItemsXMLDataResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetItemsXMLDataResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfItems"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>

```

GetItemsXMLDataResult: The container element of the **GetItemsXMLDataResult.results** element.

GetItemsXMLDataResult.results: The result of the **GetItemsXMLData** WSDL operation.

3.1.4.7.3 Complex Types

3.1.4.7.3.1 ArrayOfItems

Array of list items.

```

<s:complexType name="ArrayOfItems">
  <s:sequence>
    <s:element name="item" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="name" type="s:string"/>
        <s:attribute name="ID" type="s:unsignedInt"/>
        <s:attribute name="Author" type="s:string"/>
        <s:attribute name="Editor" type="s:string"/>
        <s:attribute name="File_x0020_Size" type="s:string"/>
        <s:attribute name="ImageWidth" type="s:unsignedInt"/>
        <s:attribute name="ImageHeight" type="s:unsignedInt"/>
        <s:attribute name="Description" type="s:string"/>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>

```

```

    <s:attribute name="Title" type="s:string"/>
    <s:attribute name="Keywords" type="s:string"/>
    <s:attribute name="ImageCreateDate" type="s:string"/>
    <s:attribute name="Created" type="s:dateTime"/>
    <s:attribute name="Modified" type="s:dateTime"/>
    <s:attribute name="found" type="s:boolean" use="optional" default="true"/>
  </s:complexType>
</s:element>
</s:sequence>
</s:complexType>

```

item: Contains information about a list item.

item.name: The name of the list item.

item.ID: The identifier of the list item.

item.Author: The identifier of the user who created the list item.

item.Editor: The identifier of the user who edited the list item.

item.File_x0020_Size: The file size, in kilobytes, of the list item if that list item is an image.

item.ImageWidth: The width, in pixels, of the list item if that list item is an image.

item.ImageHeight: The height, in pixels, of the list item if that list item is an image.

item.Description: The alternative text of the list item, if the list item is an image.

item.Title: The title of the list item, if the list item is an image.

item.Keywords: The keywords of the list item, if the list item is an image.

item.ImageCreateDate: The date and time at which the picture was taken. This is applicable only if the list item is an image.

item.Created: The date and time at which the list item was created. MUST be in UTC format.

item.Modified: The date and time at which the list item was last modified. MUST be in UTC format.

item.found: True if the list item is found; otherwise, false.

3.1.4.8 GetListItems

This WSDL operation enumerates list items in the folder in the list (1).

```

<wsdl:operation name="GetListItems">
  <wsdl:input message="GetListItemsSoapIn" />
  <wsdl:output message="GetListItemsSoapOut" />
</wsdl:operation>

```

The client sends a **GetListItemsSoapIn** request WSDL message, and the server responds with a **GetListItemsSoapOut** response WSDL message as follows:

- If **strListName** is omitted, the server MUST return an **InvalidArgument** SOAP fault.

- If the **strListName** is an empty string or does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- Otherwise, the server MUST send a **GetListItemsSoapOut** response.

3.1.4.8.1 Messages

3.1.4.8.1.1 GetListItemsSoapIn

This WSDL message contains information required by the **GetListItems** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetListItems
```

The SOAP body contains a **GetListItems** element.

3.1.4.8.1.2 GetListItemsSoapOut

This WSDL message contains results returned by the **GetListItems** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/GetListItems
```

The SOAP body contains a **GetListItemsResponse** element.

3.1.4.8.2 Elements

3.1.4.8.2.1 GetListItems

Input data for a **GetListItems** WSDL operation.

```
<s:element name="GetListItems">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST be non-empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded.

3.1.4.8.2.2 GetListItemsResponse

Result data for a GetListItems WSDL operation.

```
<s:element name="GetListItemsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetListItemsResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="Library">
              <s:complexType mixed="true">
                <s:complexContent>
                  <s:extension base="tns:ArrayOfRows" >
                    <s:attribute name="name" type="s:string"/>
                  </s:extension>
                </s:complexContent>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetListItemsResult: Contains the result of the **GetListItems** WSDL operation.

GetListItemsResult.Library: The array of list items in the folder that is specified by the **strFolder** parameter.

GetListItemsResult.Library.Name: The names of the attributes containing the list item data in this response correspond to the **Name** attribute in the **Field** elements of **GetList** and are prefixed by "ows_". Note that the set of fields returned by the method is restricted by the **viewField** or **viewName** parameter. The response includes the **ows_ServerRedirected** attribute to indicate whether server rendering is available for the item. A value of 0 indicates that no rendering mechanism is available for this item on the server. A value of 1 indicates that a rendering mechanism is available for this item on the server. If a rendering mechanism is available, the response MUST include the **ows_ServerRedirectedUrl** attribute. The value of this attribute MUST specify the link used to render the item on the server.

3.1.4.9 ListPictureLibrary

This WSDL operation lists all picture libraries on the current site (2).

```
<wsdl:operation name="ListPictureLibrary">
  <wsdl:input message="ListPictureLibrarySoapIn" />
  <wsdl:output message="ListPictureLibrarySoapOut" />
</wsdl:operation>
```

The client sends a **ListPictureLibrarySoapIn** request WSDL message, and the server MUST respond with a **ListPictureLibrarySoapOut** response WSDL message.

3.1.4.9.1 Messages

3.1.4.9.1.1 ListPictureLibrarySoapIn

This WSDL message contains information required by the **ListPictureLibrary** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/ListPictureLibrary
```

The SOAP body contains a **ListPictureLibrary** element.

3.1.4.9.1.2 ListPictureLibrarySoapOut

This WSDL message contains results returned by the **ListPictureLibrary** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/ListPictureLibrary
```

The SOAP body contains a **ListPictureLibraryResponse** element.

3.1.4.9.2 Elements

3.1.4.9.2.1 ListPictureLibrary

Input data for a **ListPictureLibrary** WSDL operation.

```
<s:element name="ListPictureLibrary">
  <s:complexType/>
</s:element>
```

3.1.4.9.2.2 ListPictureLibraryResponse

Result data for a **ListPictureLibrary** WSDL operation.

```
<s:element name="ListPictureLibraryResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="ListPictureLibraryResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="PictLib" type="tns:ArrayOfLibraries"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```


ListPictureLibraryResult: Contains the result of the **ListPictureLibrary** WSDL operation.

ListPictureLibraryResult.PictLib: The array of picture libraries.

3.1.4.9.3 Complex Types

3.1.4.9.3.1 ArrayOfLibraries

Array of libraries.

```
<s:complexType name="ArrayOfLibraries">
  <s:sequence>
    <s:element name="Library" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="guid" type="s:string"/>
        <s:attribute name="name" type="s:string"/>
        <s:attribute name="title" type="s:string"/>
        <s:attribute name="url" type="s:string"/>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>
```

Library: Contains information about a picture library.

Library.guid: The GUID of the picture library.

Library.name: The GUID of the picture library.

Library.title: The title of the picture library.

Library.url: The title of the picture library.

3.1.4.10 Rename

This WSDL operation renames list items in a list (1) and folder.

```
<wsdl:operation name="Rename">
  <wsdl:input message="RenameSoapIn" />
  <wsdl:output message="RenameSoapOut" />
</wsdl:operation>
```

The client sends a **RenameSoapIn** request WSDL message, and the server responds with a **RenameSoapOut** response WSDL message, as follows:

- If **strListName** is empty, the server MUST return an **InvalidArgument** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.

- Otherwise, the server MUST send a **RenameSoapOut** response as follows: for the list items that were both found and successfully renamed, the **renamed** flag is set; for any other list items, the **renamed** flag is not set.

3.1.4.10.1 Messages

3.1.4.10.1.1 RenameSoapIn

This WSDL message contains information required by the **Rename** WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Rename
```

The SOAP body contains a **Rename** element.

3.1.4.10.1.2 RenameSoapOut

This WSDL message contains results returned by the **Rename** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Rename
```

The SOAP body contains a **RenameResponse** element.

3.1.4.10.2 Elements

3.1.4.10.2.1 Rename

Input data for a **Rename** WSDL operation.

```
<s:element name="Rename">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
      <s:element name="request">
        <s:complexType>
          <s:sequence>
            <s:element name="files" type="tns:ArrayOfRenameFiles"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST be non-empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded.

Request.files: The array of files to rename.

3.1.4.10.2.2 RenameResponse

Result data for a **Rename** WSDL operation.

```
<s:element name="RenameResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="RenameResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfRenameResults"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

RenameResult.results: The results of the **Rename** WSDL operation.

3.1.4.10.3 Complex Types

3.1.4.10.3.1 ArrayOfRenameFiles

Array of current file names and new names, without paths.

```
<s:complexType name="ArrayOfRenameFiles">
  <s:sequence>
    <s:element name="file" maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="filename" type="s:string"/>
        <s:attribute name="newbasename" type="s:string"/>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>
```

file: Contains information about the file.

file.filename: The name of the list item, including the extension.

file.newbasename: The new name of the list item, without a path or extension.

3.1.4.10.3.2 ArrayOfRenameResults

Array of results of a **Rename** WSDL operation.

```
<s:complexType name="ArrayOfRenameResults">
  <s:sequence>
    <s:element name="result" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
```

```

        <s:attribute name="name" type="s:string"/>
        <s:attribute name="renamed" type="s:boolean" use="optional" default="true"/>
        <s:attribute name="lastmodified" type="s:dateTime"/>
        <s:attribute name="newbasename" type="s:string"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>

```

result: Contains the result of the **Rename** WSDL operation.

result.name: The name of the list item before it was renamed.

result.newbasename: The new name of the list item, without a path.

result.lastmodified: The time stamp of the last modification. MUST be in UTC format.

result.renamed: True if the **Rename** operation succeeds; otherwise, false.

3.1.4.11 Upload

This WSDL operation uploads a file to the specified list (1) and folder.

```

<wsdl:operation name="Upload">
  <wsdl:input message="UploadSoapIn" />
  <wsdl:output message="UploadSoapOut" />
</wsdl:operation>

```

The client sends an **UploadSoapIn** request WSDL message, and the server responds with an **UploadSoapOut** response WSDL message, as follows:

- If **strListName**, **bytes**, or **fileName** is omitted, the server MUST return an **InvalidArgument** SOAP fault.
- If **fileName** contains any illegal character(s) as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **IllegalFileName** SOAP fault.
- If the **strListName** does not exist, the server MUST return a **ListNotFound** SOAP fault.
- If the **strListName** is not a picture library, the server MUST return an **IsNotLibrary** SOAP fault.
- If the **strFolder** is not a legal folder path as specified in **SOAPFaultDetails** (section [2.2.4.3](#)), the server MUST return an **InvalidArgument** SOAP fault.
- If the **strFolder** does not exist, the server MUST return a **FolderNotFound** SOAP fault.
- If **fileName** already exists on the server and **fOverWriteIfExist** is set to FALSE, the server MUST return a **FileExists** SOAP fault.
- Otherwise, the server MUST send an **UploadSoapOut** response with the time stamp and upload the file as specified.

3.1.4.11.1 Messages

3.1.4.11.1.1 UploadSoapIn

This WSDL message contains information required by the Upload WSDL operation. This WSDL message is a request.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Upload
```

The SOAP body contains an Upload element.

3.1.4.11.1.2 UploadSoapOut

This WSDL message contains results returned by the **Upload** WSDL operation. This WSDL message is a response.

The SOAP action value of the WSDL message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/ois/Upload
```

The SOAP body contains an **UploadResponse** element.

3.1.4.11.2 Elements

3.1.4.11.2.1 Upload

Input data for an **Upload** WSDL operation.

```
<s:element name="Upload">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
      <s:element name="bytes" type="s:base64Binary"/>
      <s:element name="fileName" type="s:string"/>
      <s:element name="fOverWriteIfExist" type="s:boolean"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

strListName: The name of the list (1) on the current site (2). MUST be URL-encoded. MUST NOT be empty.

strFolder: The relative path from the root of the list (1) to the target folder. MUST be URL-encoded. An empty value indicates the root folder of the list (1).

bytes: The binary content of the file to upload. MUST be encoded in Base64, as specified in [\[RFC4648\]](#). MUST NOT be empty.

fileName: The file name to use as the list item name in the list (1). MUST be URL-encoded.

fOverWriteIfExists: Specifies whether to overwrite a file with the same name. This element **MUST** be present.

3.1.4.11.2.2 UploadResponse

Result data for an Upload WSDL operation.

```
<s:element name="UploadResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="UploadResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="Upload">
              <s:complexType>
                <s:attribute name="lastmodified" type="s:dateTime"/>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

UploadResult: The elements containing the result of the **Upload** WSDL operation.

UploadResult.Upload: The result of the **Upload** WSDL operation.

UploadResult.Upload.lastmodified: The time stamp, which **MUST** be in UTC format, of the last modification.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

The following examples demonstrate the interactions between the client and the server. Only the SOAP body is listed for the sake of brevity.

To create a new folder named Zoo, the client sends a **CreateNewFolderSoapIn** request to the server:

```
<CreateNewFolder xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strListName>
    Shared Pictures
  </strListName>
  <strParentFolder />
</CreateNewFolder>
```

The server confirms the WSDL operation in **CreateNewFolderSoapOut** as follows:

```
<CreateNewFolderResponse
  xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
  <CreateNewFolderResult>
    <NewFolder
      xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
      title="New folder (1)"/>
    </CreateNewFolderResult>
  </CreateNewFolderResponse>
```

The client needs to explicitly rename New folder(1) to Zoo, sending **RenameSoapIn** to the server:

```
<Rename xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strListName>
    Shared Pictures
  </strListName>
  <strFolder />
  <request>
    <files xmlns="">
      <file filename="New folder(1)" newbasename="Zoo" />
    </files>
  </request>
</Rename>
```

The server responds with **RenameSoapOut** as follows:

```
<RenameResponse xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <RenameResult>
    <results xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
      <result name="New folder (1)" renamed="true"
        newbasename="Zoo"/>
    </results>
  </RenameResult>
</RenameResponse>
```

Then, the client proceeds to upload a picture to the Zoo folder, sending **UploadSoapIn** to the server:

```
<Upload xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strListName>
    Shared Pictures
  </strListName>
  <strFolder>
    Zoo
  </strFolder>
  <bytes>
    // base64 encoded image content ...
  </bytes>
  <fileName>
    panda.jpg
  </fileName>
  <fOverWriteIfExist>
    True
  </fOverWriteIfExist>
</Upload>
```

The server confirms the upload WSDL operation in **UploadSoapOut**, as follows:

```
<UploadResponse xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <UploadResult>
    <Upload xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
      lastmodified="2008-01-18T23:44:34Z"/>
  </UploadResult>
</UploadResponse>
```

The client checks the validity of the URL by sending **CheckSubwebAndListSoapIn**, as follows:

```
<CheckSubwebAndList
  xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strUrl>
    http://site/Shared Pictures/Zoo/panda.jpg
  </strUrl>
</CheckSubwebAndList>
```

The server tries to resolve the URL and responds with **CheckSubwebAndListSoapOut**, as follows:

```
<CheckSubwebAndListResponse
  xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <CheckSubwebAndListResult>
    <result xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
      url="http://site/Shared Pictures/Zoo/panda.jpg"
      subweb="http://site" list="Shared Pictures"
      listGuid="502939ee-8e57-43d3-a5d8-e19fe90313ce" folder="Zoo"
      rest="/panda.jpg" />
  </CheckSubwebAndListResult>
</CheckSubwebAndListResponse>
```


Then, the client requests data on all the pictures in the Zoo folder, using **GetListItemsSoapIn**, as follows:

```
<GetListItems xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strListName>
    Shared Pictures
  </strListName>
  <strFolder>
    Zoo
  </strFolder>
</GetListItems>
```

The following is the **GetListItemsSoapOut** response from the server:

```
<GetListItemsResponse
  xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
  <GetListItemsResult>
    <Library xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
      name="Shared Pictures">
      <z:row xmlns:z="#RowsetSchema" ows_ID='75' ows_Author='1;#Ye
        Xu' ows_Editor='1;#Ye Xu' ows_Created='2008-01-18T23:44:34Z'
        ows_Modified='2008-01-18T23:44:34Z'
        ows_File_x0020_Size='75;#1000' ows_FSObjType='75;#0'
        ows_FileLeafRef='75;#panda.jpg'
        ows_EncodedAbsUrl='http://site/Shared%20Pictures/Zoo/panda.jpg'
        ows_ImageWidth='15' ows_ImageHeight='20'
        ows_ModerationStatus='0' ows_Level='1'
        ows_UniqueId='75;#{B7AE51A7-9503-40E7-9B82-F830DEC2CB6B}'
        ows_Created_x0020_Date='75;#2008-01-18T23:44:34Z'
        ows_ProgId='75;#' ows_FileRef='75;#sites/Shared
        Pictures/Zoo/panda.jpg' ows_DocIcon='jpg'

        ows_MetaInfo='75;#vti_parserversion:SR|14.0.0.4730..vti_modifiedby:SR|CONTOSO\yexu..vti_lasth
        eight:IX|20..ContentTypeId:SW|0x01010200BC73A9F9558BC445A09D0F26FB930A03..vti_lastwidth:IX|15
        ..vti_author:SR| CONTOSO\yexu..'
        ows_Last_x0020_Modified='75;#2008-01-18T23:44:34Z'
        ows_owshiddenversion='1' ows_ServerRedirected='0'/'>
      <z:row xmlns:z="#RowsetSchema" ows_ID='76' ows_Author='1;#Ye
        Xu' ows_Editor='1;#Ye Xu' ows_Created='2009-01-18T23:44:35Z'
        ows_Modified='2009-01-18T23:44:35Z'
        ows_File_x0020_Size='76;#1000' ows_FSObjType='76;#0'
        ows_FileLeafRef='76;#dinosaur.jpg'
        ows_EncodedAbsUrl='http://site/Shared%20Pictures/Zoo/dinosaur.jpg'
        ows_ImageWidth='15' ows_ImageHeight='20'
        ows_ModerationStatus='0' ows_Level='1'
        ows_UniqueId='76;#{4691F195-1513-477F-9EB4-AABD2C328259}'
        ows_Created_x0020_Date='76;#2009-01-18T23:44:35Z'
        ows_ProgId='76;#' ows_FileRef='76;#sites/Shared
        Pictures/Zoo/dinosaur.jpg' ows_DocIcon='jpg'

        ows_MetaInfo='76;#vti_parserversion:SR|14.0.0.4730..vti_modifiedby:SR|CONTOSO\yexu..vti_lasth
        eight:IX|20..ContentTypeId:SW|0x01010200BC73A9F9558BC445A09D0F26FB930A03..vti_lastwidth:IX|15
        ..vti_author:SR| CONTOSO\yexu..'
        ows_Last_x0020_Modified='76;#2009-01-18T23:44:35Z'
        ows_owshiddenversion='1' ows_ServerRedirected='0'/'>
    </Library>
  </GetListItemsResult>
```

```
</GetListItemsResponse>
```

Later, to build an encyclopedia, the client downloads some images from the server as illustrations, sending **DownloadSoapIn** to the server, as follows:

```
<Download xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <strListName>
    Shared Pictures
  </strListName>
  <strFolder>
    Zoo
  </strFolder>
  <itemFileNames>
    <string>
      panda.jpg
    </string>
    <string>
      dinosaur.jpg
    </string>
  </itemFileNames>
  <type>
    0
  </type>
  <fetchOriginalIfNotAvailable>
    true
  </fetchOriginalIfNotAvailable>
</Download>
```

Dinosaur is not found in the Zoo folder, so the server responds with **DownloadSoapOut**, as follows:

```
<DownloadResponse
  xmlns="http://schemas.microsoft.com/sharepoint/soap/ois/"
  <DownloadResult>
    <Files>
      <File name="panda.jpg" lastmodified="2008-01-16T23:12:57Z">
        ... ..
      </File>
      <File name="dinosaur.jpg" found="false"/>
    </Files>
  </DownloadResult>
</DownloadResponse>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL is provided below:

```
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/soap/ois/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/ois/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/ois/">
      <s:import namespace="http://www.w3.org/2001/XMLSchema"/>
      <s:element name="ListPictureLibrary">
        <s:complexType/>
      </s:element>
      <s:element name="ListPictureLibraryResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="ListPictureLibraryResult">
              <s:complexType mixed="true">
                <s:sequence>
                  <s:element name="PictLib" type="tns:ArrayOfLibraries"/>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfLibraries">
        <s:sequence>
          <s:element name="Library" minOccurs="0" maxOccurs="unbounded">
            <s:complexType>
              <s:attribute name="guid" type="s:string"/>
              <s:attribute name="name" type="s:string"/>
              <s:attribute name="title" type="s:string"/>
              <s:attribute name="url" type="s:string"/>
            </s:complexType>
          </s:element>
        </s:sequence>
      </s:complexType>
      <s:element name="Upload">
        <s:complexType>
          <s:sequence>
            <s:element name="strListName" type="s:string"/>
            <s:element name="strFolder" type="s:string"/>
            <s:element name="bytes" type="s:base64Binary"/>
            <s:element name="fileName" type="s:string"/>
            <s:element name="fOverWriteIfExist" type="s:boolean"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="UploadResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="UploadResult">
              <s:complexType mixed="true">
                <s:sequence>
```

```

        <s:element name="Upload">
            <s:complexType>
                <s:attribute name="lastmodified" type="s:dateTime"/>
            </s:complexType>
        </s:element>
    </s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
<s:element name="GetItemsXMLData">
    <s:complexType>
        <s:sequence>
            <s:element name="strListName" type="s:string"/>
            <s:element name="strFolder" type="s:string"/>
            <s:element name="itemFileNames" type="tns:ArrayOfString"/>
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfString">
    <s:sequence>
        <s:element name="string" type="s:string" maxOccurs="unbounded"/>
    </s:sequence>
</s:complexType>
<s:element name="GetItemsXMLDataResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="GetItemsXMLDataResult">
                <s:complexType mixed="true">
                    <s:sequence>
                        <s:element name="results" type="tns:ArrayOfItems"/>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfItems">
    <s:sequence>
        <s:element name="item" minOccurs="0" maxOccurs="unbounded">
            <s:complexType>
                <s:attribute name="name" type="s:string"/>
                <s:attribute name="ID" type="s:unsignedInt"/>
                <s:attribute name="Author" type="s:string"/>
                <s:attribute name="Editor" type="s:string"/>
                <s:attribute name="File_x0020_Size" type="s:string"/>
                <s:attribute name="ImageWidth" type="s:unsignedInt"/>
                <s:attribute name="ImageHeight" type="s:unsignedInt"/>
                <s:attribute name="Description" type="s:string"/>
                <s:attribute name="Title" type="s:string"/>
                <s:attribute name="Keywords" type="s:string"/>
                <s:attribute name="ImageCreateDate" type="s:string"/>
                <s:attribute name="Created" type="s:dateTime"/>
                <s:attribute name="Modified" type="s:dateTime"/>
                <s:attribute name="found" type="s:boolean" use="optional" default="true"/>
            </s:complexType>
        </s:element>
    </s:sequence>

```

```

</s:complexType>
<s:element name="GetItemsByIds">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="ids" type="tns:ArrayOfUnsignedInt"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfUnsignedInt">
  <s:sequence>
    <s:element name="unsignedInt" type="s:unsignedInt" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
<s:element name="GetItemsByIdsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetItemsByIdsResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfRows"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfRows" mixed="true">
  <s:sequence>
    <s:any minOccurs="0" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
<s:element name="Delete">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
      <s:element name="itemFileNames" type="tns:ArrayOfString"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="DeleteResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="DeleteResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="results" type="tns:ArrayOfDeleteResults"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfDeleteResults">
  <s:sequence>
    <s:element name="result" minOccurs="0" maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="deleted" type="s:boolean"/>

```

```

        <s:attribute name="name" type="s:string"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>
<s:element name="Download">
    <s:complexType>
        <s:sequence>
            <s:element name="strListName" type="s:string"/>
            <s:element name="strFolder" type="s:string"/>
            <s:element name="itemFileNames" type="tns:ArrayOfString"/>
            <s:element name="type" type="s:unsignedInt"/>
            <s:element name="fFetchOriginalIfNotAvailable" type="s:boolean"/>
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="DownloadResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="DownloadResult">
                <s:complexType mixed="true">
                    <s:sequence>
                        <s:element name="Files" type="tns:ArrayOfFiles"/>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfFiles">
    <s:sequence>
        <s:element name="File" minOccurs="0" maxOccurs="unbounded">
            <s:complexType>
                <s:simpleContent>
                    <s:extension base="s:base64Binary">
                        <s:attribute name="name" type="s:string"/>
                        <s:attribute name="lastmodified" type="s:dateTime"/>
                        <s:attribute name="found" type="s:boolean" use="optional" default="true"/>
                    </s:extension>
                </s:simpleContent>
            </s:complexType>
        </s:element>
    </s:sequence>
</s:complexType>
<s:element name="Edit">
    <s:complexType>
        <s:sequence>
            <s:element name="strListName" type="s:string" minOccurs="0"/>
            <s:element name="strFolder" type="s:string" minOccurs="0"/>
            <s:element name="itemFileName" type="s:string" minOccurs="0"/>
            <s:element name="recipe" minOccurs="0">
                <s:complexType>
                    <s:sequence>
                        <s:any/>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>

```

```

</s:element>
<s:element name="EditResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="EditResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="notImplemented"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetListItems">
  <s:complexType>
    <s:sequence>
      <s:element name="strListName" type="s:string"/>
      <s:element name="strFolder" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetListItemsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetListItemsResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="Library">
              <s:complexType mixed="true">
                <s:complexContent>
                  <s:extension base="tns:ArrayOfRows">
                    <s:attribute name="name" type="s:string"/>
                  </s:extension>
                </s:complexContent>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="CheckSubwebAndList">
  <s:complexType>
    <s:sequence>
      <s:element name="strUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="CheckSubwebAndListResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="CheckSubwebAndListResult">
        <s:complexType mixed="true">
          <s:sequence>
            <s:element name="result">
              <s:complexType>
                <s:attribute name="url" type="s:string"/>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>

```



```

        <s:attribute name="subweb" type="s:string"/>
        <s:attribute name="list" type="s:string"/>
        <s:attribute name="listGuid" type="s:string"/>
        <s:attribute name="folder" type="s:string"/>
        <s:attribute name="rest" type="s:string"/>
        <s:attribute name="found" type="s:boolean" use="optional"
default="true"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
<s:element name="Rename">
    <s:complexType>
        <s:sequence>
            <s:element name="strListName" type="s:string"/>
            <s:element name="strFolder" type="s:string"/>
            <s:element name="request">
                <s:complexType>
                    <s:sequence>
                        <s:element name="files" type="tns:ArrayOfRenameFiles"/>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfRenameFiles">
    <s:sequence>
        <s:element name="file" maxOccurs="unbounded">
            <s:complexType>
                <s:attribute name="filename" type="s:string"/>
                <s:attribute name="newbasename" type="s:string"/>
            </s:complexType>
        </s:element>
    </s:sequence>
</s:complexType>
<s:element name="RenameResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="RenameResult">
                <s:complexType mixed="true">
                    <s:sequence>
                        <s:element name="results" type="tns:ArrayOfRenameResults"/>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfRenameResults">
    <s:sequence>
        <s:element name="result" minOccurs="0" maxOccurs="unbounded">
            <s:complexType>
                <s:attribute name="name" type="s:string"/>
                <s:attribute name="renamed" type="s:boolean" use="optional" default="true"/>

```

```

        <s:attribute name="lastmodified" type="s:dateTime"/>
        <s:attribute name="newbasename" type="s:string"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>
<s:element name="CreateNewFolder">
    <s:complexType>
        <s:sequence>
            <s:element name="strListName" type="s:string"/>
            <s:element name="strParentFolder" type="s:string"/>
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="CreateNewFolderResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="CreateNewFolderResult">
                <s:complexType mixed="true">
                    <s:sequence>
                        <s:element name="NewFolder">
                            <s:complexType>
                                <s:attribute name="title" type="s:string"/>
                            </s:complexType>
                        </s:element>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="ListPictureLibrarySoapIn">
    <wsdl:part name="parameters" element="tns:ListPictureLibrary"/>
</wsdl:message>
<wsdl:message name="ListPictureLibrarySoapOut">
    <wsdl:part name="parameters" element="tns:ListPictureLibraryResponse"/>
</wsdl:message>
<wsdl:message name="UploadSoapIn">
    <wsdl:part name="parameters" element="tns:Upload"/>
</wsdl:message>
<wsdl:message name="UploadSoapOut">
    <wsdl:part name="parameters" element="tns:UploadResponse"/>
</wsdl:message>
<wsdl:message name="GetItemsXMLDataSoapIn">
    <wsdl:part name="parameters" element="tns:GetItemsXMLData"/>
</wsdl:message>
<wsdl:message name="GetItemsXMLDataSoapOut">
    <wsdl:part name="parameters" element="tns:GetItemsXMLDataResponse"/>
</wsdl:message>
<wsdl:message name="GetItemsByIdsSoapIn">
    <wsdl:part name="parameters" element="tns:GetItemsByIds"/>
</wsdl:message>
<wsdl:message name="GetItemsByIdsSoapOut">
    <wsdl:part name="parameters" element="tns:GetItemsByIdsResponse"/>
</wsdl:message>
<wsdl:message name="DeleteSoapIn">
    <wsdl:part name="parameters" element="tns:Delete"/>

```

```

</wsdl:message>
<wsdl:message name="DeleteSoapOut">
  <wsdl:part name="parameters" element="tns:DeleteResponse"/>
</wsdl:message>
<wsdl:message name="DownloadSoapIn">
  <wsdl:part name="parameters" element="tns:Download"/>
</wsdl:message>
<wsdl:message name="DownloadSoapOut">
  <wsdl:part name="parameters" element="tns:DownloadResponse"/>
</wsdl:message>
<wsdl:message name="EditSoapIn">
  <wsdl:part name="parameters" element="tns:Edit"/>
</wsdl:message>
<wsdl:message name="EditSoapOut">
  <wsdl:part name="parameters" element="tns:EditResponse"/>
</wsdl:message>
<wsdl:message name="GetListItemsSoapIn">
  <wsdl:part name="parameters" element="tns:GetListItems"/>
</wsdl:message>
<wsdl:message name="GetListItemsSoapOut">
  <wsdl:part name="parameters" element="tns:GetListItemsResponse"/>
</wsdl:message>
<wsdl:message name="CheckSubwebAndListSoapIn">
  <wsdl:part name="parameters" element="tns:CheckSubwebAndList"/>
</wsdl:message>
<wsdl:message name="CheckSubwebAndListSoapOut">
  <wsdl:part name="parameters" element="tns:CheckSubwebAndListResponse"/>
</wsdl:message>
<wsdl:message name="RenameSoapIn">
  <wsdl:part name="parameters" element="tns:Rename"/>
</wsdl:message>
<wsdl:message name="RenameSoapOut">
  <wsdl:part name="parameters" element="tns:RenameResponse"/>
</wsdl:message>
<wsdl:message name="CreateNewFolderSoapIn">
  <wsdl:part name="parameters" element="tns:CreateNewFolder"/>
</wsdl:message>
<wsdl:message name="CreateNewFolderSoapOut">
  <wsdl:part name="parameters" element="tns:CreateNewFolderResponse"/>
</wsdl:message>
<wsdl:portType name="ImagingSoap">
  <wsdl:operation name="ListPictureLibrary">
    <wsdl:input message="tns:ListPictureLibrarySoapIn"/>
    <wsdl:output message="tns:ListPictureLibrarySoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="Upload">
    <wsdl:input message="tns:UploadSoapIn"/>
    <wsdl:output message="tns:UploadSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="GetItemsXMLData">
    <wsdl:input message="tns:GetItemsXMLDataSoapIn"/>
    <wsdl:output message="tns:GetItemsXMLDataSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="GetItemsByIds">
    <wsdl:input message="tns:GetItemsByIdsSoapIn"/>
    <wsdl:output message="tns:GetItemsByIdsSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="Delete">
    <wsdl:input message="tns:DeleteSoapIn"/>

```

```

        <wsdl:output message="tns:DeleteSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="Download">
        <wsdl:input message="tns:DownloadSoapIn"/>
        <wsdl:output message="tns:DownloadSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="Edit">
        <wsdl:input message="tns:EditSoapIn"/>
        <wsdl:output message="tns:EditSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="GetListItems">
        <wsdl:input message="tns:GetListItemsSoapIn"/>
        <wsdl:output message="tns:GetListItemsSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="CheckSubwebAndList">
        <wsdl:input message="tns:CheckSubwebAndListSoapIn"/>
        <wsdl:output message="tns:CheckSubwebAndListSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="Rename">
        <wsdl:input message="tns:RenameSoapIn"/>
        <wsdl:output message="tns:RenameSoapOut"/>
    </wsdl:operation>
    <wsdl:operation name="CreateNewFolder">
        <wsdl:input message="tns:CreateNewFolderSoapIn"/>
        <wsdl:output message="tns:CreateNewFolderSoapOut"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ImagingSoap" type="tns:ImagingSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="ListPictureLibrary">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/ListPictureLibrary"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Upload">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Upload"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetItemsXMLData">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsXMLData"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetItemsByIds">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsByIds"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Delete">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Delete"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Download">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Download"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Edit">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Edit"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetListItems">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetListItems" style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CheckSubwebAndList">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/CheckSubwebAndList"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
    </wsdl:operation>

```

```

        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Rename">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Rename"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateNewFolder">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/CreateNewFolder"
style="document"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="ImagingSoap12" type="tns:ImagingSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="ListPictureLibrary">
        <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/ListPictureLibrary"
style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Upload">
        <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Upload"
style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetItemsXMLData">
        <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsXMLData"
style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

    <wsdl:operation name="GetItemsByIds">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetItemsByIds"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Delete">
      <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Delete"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Download">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Download" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Edit">
      <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Edit"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetListItems">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/GetListItems" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CheckSubwebAndList">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/CheckSubwebAndList"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

```

```

        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Rename">
        <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/Rename"
style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreateNewFolder">
        <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ois/CreateNewFolder"
style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```


7 Appendix B: Product Behavior

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

- Microsoft® Office 2003
- The 2007 Microsoft® Office system
- Microsoft® Office 2010 suites
- Microsoft® SharePoint® Foundation 2010
- Windows® SharePoint® Services 3.0

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

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

8 Change Tracking

This section identifies changes that were made to the [MS-IMAGS] protocol document between the March 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

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

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

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

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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

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

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.5 Prerequisites/Preconditions	Updated the description of the protocol server endpoint.	N	Content updated.
2.2.4.3 SOAPFaultDetails	Added error code for illegal character in file name condition.	N	New content added.

9 Index

A

Abstract data model
 [server](#) 13
[Applicability](#) 9
[ArrayOfRows complex type](#) 11
[ArrayOfString complex type](#) 11
[Attribute groups](#) 12
[Attributes](#) 12

C

[Capability negotiation](#) 9
[Change tracking](#) 58
Client
 [overview](#) 13
[Complex types](#) 10
 [ArrayOfRows](#) 11
 [ArrayOfString](#) 11
 [SOAPFaultDetails](#) 11

D

Data model - abstract
 [server](#) 13

E

Events
 [local - server](#) 38
 [timer - server](#) 38
Examples
 [overview](#) 39

F

[Fields - vendor-extensible](#) 9
[Full WSDL](#) 44

G

[Glossary](#) 6
[Groups](#) 12

I

[Implementer - security considerations](#) 43
[Index of security parameters](#) 43
[Informative references](#) 7
Initialization
 [server](#) 13
[Introduction](#) 6

L

Local events
 [server](#) 38

M

Message processing
 [server](#) 13
Messages
 [ArrayOfRows complex type](#) 11
 [ArrayOfString complex type](#) 11
 [attribute groups](#) 12
 [attributes](#) 12
 [complex types](#) 10
 [elements](#) 10
 [enumerated](#) 10
 [groups](#) 12
 [namespaces](#) 10
 [simple types](#) 12
 [SOAPFaultDetails complex type](#) 11
 [syntax](#) 10
 [transport](#) 10

N

[Namespaces](#) 10
[Normative references](#) 6

O

Operations
 [CheckSubwebAndList](#) 14
 [CreateNewFolder](#) 16
 [Delete](#) 18
 [Download](#) 20
 [Edit](#) 23
 [GetItemsByIds](#) 24
 [GetItemsXMLData](#) 26
 [GetListItems](#) 29
 [ListPictureLibrary](#) 31
 [Rename](#) 33
 [Upload](#) 36
[Overview \(synopsis\)](#) 7

P

[Parameters - security index](#) 43
[Preconditions](#) 9
[Prerequisites](#) 9
[Product behavior](#) 57

R

References
 [informative](#) 7
 [normative](#) 6
[Relationship to other protocols](#) 8

S

Security

- [implementer considerations](#) 43
- [parameter index](#) 43
- Sequencing rules
 - [server](#) 13
- Server
 - [abstract data model](#) 13
 - [CheckSubwebAndList operation](#) 14
 - [CreateNewFolder operation](#) 16
 - [Delete operation](#) 18
 - [Download operation](#) 20
 - [Edit operation](#) 23
 - [GetItemsByIds operation](#) 24
 - [GetItemsXMLData operation](#) 26
 - [GetListItems operation](#) 29
 - [initialization](#) 13
 - [ListPictureLibrary operation](#) 31
 - [local events](#) 38
 - [message processing](#) 13
 - [overview](#) 13
 - [Rename operation](#) 33
 - [sequencing rules](#) 13
 - [timer events](#) 38
 - [timers](#) 13
 - [Upload operation](#) 36
- [Simple types](#) 12
- [SOAPFaultDetails complex type](#) 11
- [Standards assignments](#) 9
- Syntax
 - [messages - overview](#) 10

T

- Timer events
 - [server](#) 38
- Timers
 - [server](#) 13
- [Tracking changes](#) 58
- [Transport](#) 10
- Types
 - [complex](#) 10
 - [simple](#) 12

V

- [Vendor-extensible fields](#) 9
- [Versioning](#) 9

W

- [WSDL](#) 44