[MS-SLIDELI]:

Slide Library Web Service Protocol

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 Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **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
4/4/2008	0.1	New	Initial Availability
6/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
7/13/2009	1.02	Major	Changes made for template compliance
8/28/2009	1.03	Editorial	Revised and edited the technical content
11/6/2009	1.04	Editorial	Revised and edited the technical content
2/19/2010	2.0	Editorial	Revised and edited the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Editorial	Revised and edited the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.05	Minor	Clarified the meaning of the technical content.
9/27/2010	2.05	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.06	Editorial	Changed language and formatting in the technical content.
12/17/2010	2.06	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.06	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.06	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	3.0	Major	Significantly changed the technical content.
4/11/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	3.1	Minor	Clarified the meaning of the technical content.
9/12/2012	3.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	3.2	Minor	Clarified the meaning of the technical content.
2/11/2013	3.2	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	3.3	Minor	Clarified the meaning of the technical content.
11/18/2013	3.3	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	3.3	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
4/30/2014	3.4	Minor	Clarified the meaning of the technical content.
7/31/2014	3.4	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	3.4	None	No changes to the meaning, language, or formatting of the technical content.
3/16/2015	4.0	Major	Significantly changed the technical content.
2/26/2016	5.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Slide Library Web Service Protocol enables a protocol client to obtain information about slides on a protocol server.

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

1.1 Glossary

This document uses the following terms:

checked out: A publishing level that indicates that a document has been created and locked for exclusive editing by a user in a version control system.

document: An object in a content database such as a file, folder, list (1), or site (2). Each object is identified by a URI.

folder: A file system construct. File systems organize a volume's data by providing a hierarchy of objects, which are referred to as folders or directories, that contain files and can also contain other folders.

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

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

language code identifier (LCID): A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.

slide: A frame that contains text, shapes, pictures, or other content. A slide is a digital equivalent to a traditional film slide.

Slide Library: A type of a document library that is optimized for storing and reusing presentation slides that conform to the format described in [ISO/IEC-29500:2008].

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

SOAP action: The HTTP request header field used to indicate the intent of the **SOAP** request, using a URI value. See [SOAP1.1] section 6.1.1 for more information.

SOAP body: A container for the payload data being delivered by a **SOAP message** to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

SOAP fault: A container for error and status information within a **SOAP message**. See [SOAP1.2-1/2007] section 5.4 for more information.

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

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

universally unique identifier (UUID): A 128-bit value. UUIDs can be used for multiple purposes, from tagging objects with an extremely short lifetime, to reliably identifying very persistent objects in cross-process communication such as client and server interfaces, manager entry-point vectors, and RPC objects. UUIDs are highly likely to be unique. UUIDs are also known as globally unique identifiers (GUIDs) and these terms are used interchangeably in the Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the UUID. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the UUID.

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

website: A group of related pages and data within a SharePoint site collection. The structure and content of a site is based on a site definition. Also referred to as SharePoint site and site.

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

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

XML namespace prefix: An abbreviated form of an **XML namespace**, as described in [XML].

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

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

1.2 References

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

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[ISO-8601] International Organization for Standardization, "Data Elements and Interchange Formats - Information Interchange - Representation of Dates and Times", ISO/IEC 8601:2004, December 2004,

http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=40874&ICS1=1&ICS2=140&ICS3=30

Note There is a charge to download the specification.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.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.rfc-editor.org/rfc/rfc2616.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

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

1.2.2 Informative References

[MS-AUTHWS] Microsoft Corporation, "Authentication Web Service Protocol".

1.3 Overview

This protocol enables a protocol client to send a request to the protocol server and then to receive from the protocol server information about the existence of named slides, information about all slides, information about slides specified by their identifiers, or information about slides that contain specific search strings.

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:

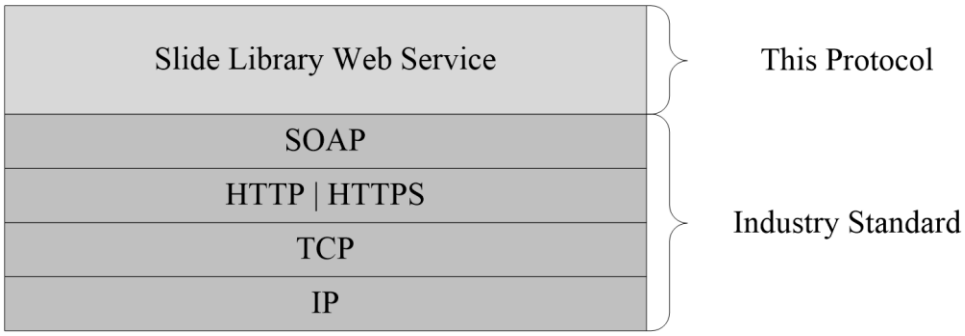

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a **website** that is identified by a **URL** that is known by protocol clients. The protocol server endpoint is formed by appending "/_vti_bin/SlideLibrary.asmx" to the URL of the Web site, for example http://www.contoso.com/Repository/_vti_bin/SlideLibrary.asmx.

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

1.6 Applicability Statement

This protocol is designed to retrieve information about **slides** that are stored on the protocol server.

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

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

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 either using HTTP status codes, as specified in [RFC2616] section 10 or using **SOAP faults**, as specified either in [SOAP1.1] section 4.4 or in [SOAP1.2/1] section 5.4.

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses **XML schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL, as specified in [WSDL].

2.2.1 Namespaces

This specification uses the following **XML namespaces** (see [XMLNS] for information about XML namespaces). Although this specification associates a specific **XML namespace prefix** with each XML namespace that it uses, the choice of any particular XML namespace prefix is an encoding detail and not significant for interoperability. These namespaces are described in the following table.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[WSDL]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[WSDL]
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1]
tns	http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

2.2.2 Messages

The following table summarizes common WSDL messages defined by this specification.

Message	Description
SOAP fault	A response message to carry the server-side exception as specified in section 4.4 of [SOAP1.1]

2.2.3 Elements

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

Element	Description
strListUrl	A string that represents the URL of a **folder** in a **Slide Library** on the protocol server.

2.2.3.1 strListUrl

This element is a string that represents the URL of a folder in a Slide Library on the protocol server.

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 operation are described with the operation.

Complex type	Description
ArrayOfString	A list of strings.
CT_Result	The result of a Slide Library Web Service operation.
CT_SlideResult	A child element of **CT_Result** that contains information about a slide.

2.2.4.1 ArrayOfString

This complex type represents a list of strings.

```
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
              type="s:string" />
  </s:sequence>
</s:complexType>
```

string: Each element MUST represent a string.

2.2.4.2 CT_Result

This complex type represents the result in response to a Slide Library Web Service request.

```
<s:complexType name="CT_Result">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="Title" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="Url" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="ForceCheckout"
               type="tns:ST TrueFalse" />
    <s:element minOccurs="0" maxOccurs="unbounded" name="Slide"
               type="tns:CT_SlideResult" />
  </s:sequence>
</s:complexType>
```

Title: This element represents the title of the Slide Library.

Url: This element represents the URL of the Slide library.

ForceCheckout: This element represents information about the slides that are **checked out** as a side effect of the operation. If **ForceCheckout** is set to "true" or "True", the slides are checked out. Otherwise, if **ForceCheckout** is set to "false" or "False", the slides are not checked out. If this element is absent, the value "false" MUST be assumed.

Slide: This element represents information about each slide to be returned.

2.2.4.3 CT_SlideResult

This complex type represents the child element of **CT_Result**, which is the result of a Slide Library Web Service operation for a slide.

```
<s:complexType name="CT_SlideResult">
  <s:attribute use="optional" name="FileName" type="s:string" />
  <s:attribute use="optional" name="Description" type="s:string" />
  <s:attribute use="optional" name="Editor" type="s:string" />
  <s:attribute use="optional" name="EncodedAbsThumbnailUrl" type="s:string" />
  <s:attribute use="required" name="Modified" type="s:string" />
  <s:attribute use="required" name="ID" type="s:unsignedInt" />
  <s:attribute use="required" name="EncodedAbsUrl" type="s:string" />
  <s:attribute use="optional" name="Presentation" type="s:string" />
  <s:attribute use="optional" name="UniqueId" type="s:string" />
</s:complexType>
```

FileName: Specifies everything after the last slash in the **EncodedAbsUrl** for a slide.

Description: Specifies the description of the slide.

Editor: Specifies the editor who last modified the slide.

EncodedAbsThumbnailUrl: Specifies the URL to a thumbnail of the slide.

Modified: Specifies the string representation of the time the slide was last modified. The value MUST be in [ISO-8601] format.

ID: Specifies the unique identifier of the slide in a Slide Library.

EncodedAbsUrl: Specifies the URL of the slide **document**.

Presentation: Specifies the name of the document from which the slide originated.

UniqueId: Specifies the unique identifier of the slide maintained by the protocol server. The scope of this identifier is within the protocol server. The value MUST be the string representation of a **UUID**.

2.2.5 Simple Types

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

Simple type	Description
ST_TrueFalse	A string type that allows a set of possible values.

2.2.5.1 ST_TrueFalse

This simple type represents a string that MUST be from a list of allowed values.

```
<s:simpleType name="ST_TrueFalse">
   <s:restriction base="s:string">
      <s:enumeration value="true"/>
      <s:enumeration value="false"/>
      <s:enumeration value="True"/>
      <s:enumeration value="False"/>
   </s:restriction>
</s:simpleType>
```

The list of allowed string values is specified by the following table.

Allowed Value
True
False
true
false

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

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

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

3.1 SlideLibrarySoap Server Details

The following diagram describes the communication between the protocol client and the protocol server.

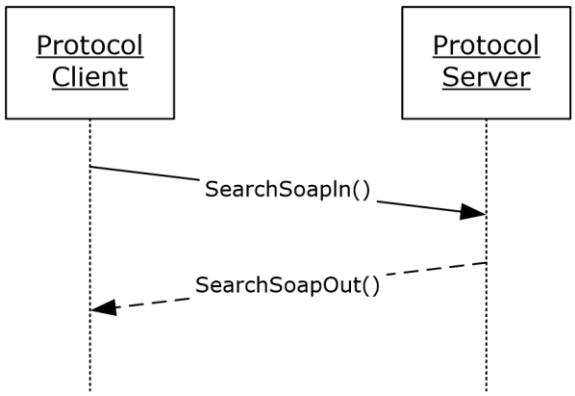

Figure 2: Sample communication between protocol client and protocol server

The protocol client sends a **SearchSoapIn** SOAP message to the protocol server.

The protocol server responds by sending a **SearchSoapOut** SOAP message to the protocol client.

3.1.1 Abstract Data Model

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

The protocol server maintains a list of files which can be referenced uniquely by URL. The files contain a presentation application file that has a single slide, and have associated metadata as fields.

The following diagram shows the abstract data model.

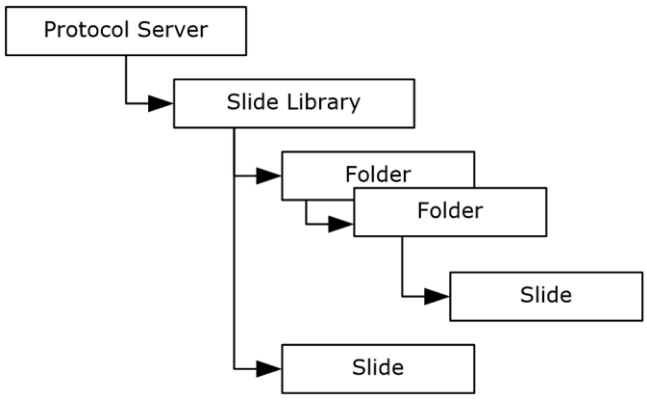

Figure 3: Abstract data model for Slide Library

3.1.2 Timers

None.

3.1.3 Initialization

The protocol server MUST expose its Web methods at the following URL, which builds upon a base URL. The URL MUST conform to the following structure: *base URL*/_vti_bin/SlideLibrary.asmx. This is the minimal required structure. Case-sensitivity is specific to the protocol server implementation.

3.1.4 Message Processing Events and Sequencing Rules

Operation	Description
CheckCollisions	This operation is used to check the existence of named slides in a folder of a Slide Library.
GetSlidesXML	This operation is used to get information about all slides in a folder of a slide library.
GetSlideInfoByIds	This operation is used to get information about slides specified by their identifiers in a folder of a slide library.
Search	This operation is used to search for slides that contain the specified search string in a folder of slide library.

3.1.4.1 CheckCollisions

The **CheckCollisions** operation is used to check the existence of named slides in a folder of Slide Library.

```
<wsdl:operation name="CheckCollisions">
      <wsdl:input message="tns:CheckCollisionsSoapIn" />
      <wsdl:output message="tns:CheckCollisionsSoapOut" />
</wsdl:operation>
```

The protocol client sends a **CheckCollisionsSoapIn** request message and the protocol server responds with a **CheckCollisionsSoapOut** response message.

3.1.4.1.1 Messages

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

Message	Description
CheckCollisionsSoapIn	The request message for checking the existence of named slides in the Slide Library.
CheckCollisionsSoapOut	The response message to a **CheckCollisionsSoapIn** request message.

3.1.4.1.1.1 CheckCollisionsSoapIn

This is the request message for checking the existence of named slides in the Slide Library.

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

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/CheckCollisions
```

The **SOAP body** MUST contain a **CheckCollisions** element.

3.1.4.1.1.2 CheckCollisionsSoapOut

The **CheckCollisionsSoapOut** message is the response to a **CheckCollisionsSoapIn** request message.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/CheckCollisions
```

The SOAP body MUST contain a **CheckCollisionsResponse** element.

3.1.4.1.2 Elements

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

Element	Description
CheckCollisions	Specifies the input parameters for a **CheckCollisions** operation.
CheckCollisionsResponse	Specifies the output of a **CheckCollisions** operation.

3.1.4.1.2.1 CheckCollisions

The **CheckCollisions** element specifies the input parameters for a **CheckCollisions** operation.

```
<s:element name="CheckCollisions">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="slideNames"
                 type="tns:ArrayOfString" />
    </s:sequence>
  </s:complexType>
</s:element>
```

strListUrl: This element MUST be present, as specified in section 2.2.3.1.

slideNames: This element specifies a list of strings, where each string represents the file name of a slide to be checked.

If **slideNames** is present, each **string** MUST NOT be empty and MUST NOT contain any of the following characters:

- Double backslash (\\)
- Colon (:)
- Asterisk (*)
- Question mark (?)
- Apostrophe (')
- Left angle bracket (<)
- Right angle bracket (>)
- Vertical bar (|)
- Number sign (#)
- \t
- Left curly brace ({)
- Right curly brace (})
- Percent sign (%)

3.1.4.1.2.2 CheckCollisionsResponse

The **CheckCollisionsResponse** element specifies the output of a **CheckCollisions** operation.

```
<s:element name="CheckCollisionsResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="CheckCollisionsResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
                       type="tns:CT Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

CheckCollisionsResult: This element specifies the result of the **CheckCollisions** operation.

CheckCollisionsResult.results: This element specifies the result of the **CheckCollisions** operation.

If no slides exist with the file names specified in **slideNames**, the protocol server MUST return a fault. Otherwise, the protocol server MUST send a **CheckCollisionsSoapOut** message.

If a **string** is a file name for a slide in the folder of Slide Library, the **CheckCollisionsSoapOut** message MUST include a single **results** element with a **Slide** element for each of the slides, and the **Slide** element MUST contain the **FileName** attribute.

3.1.4.2 GetSlidesXML

The **GetSlidesXML** operation is used to get the information for all slides in a folder of the Slide Library.

```
<wsdl:operation name="GetSlidesXML">
      <wsdl:input message="tns:GetSlidesXMLSoapIn" />
      <wsdl:output message="tns:GetSlidesXMLSoapOut" />
</wsdl:operation>
```

The protocol client sends a **GetSlidesXMLSoapIn** request message, and the protocol server responds with a **GetSlidesXMLSoapOut** response message.

3.1.4.2.1 Messages

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

Message	Description
GetSlideXMLSoapIn	The request message for getting information about slides in a Slide Library.
GetSlideXMLSoapOut	The response message to a **GetSlidesXMLSoapIn** request message.

3.1.4.2.1.1 GetSlidesXMLSoapIn

This is the request message for getting information about slides in a Slide Library.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlidesXML
```

The SOAP body MUST contain a **GetSlidesXML** element.

3.1.4.2.1.2 GetSlidesXMLSoapOut

This is the response message to a **GetSlidesXMLSoapIn** request message.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlidesXML
```

The SOAP body MUST contain a **GetSlidesXMLResponse** element.

3.1.4.2.2 Elements

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

Element	Description
GetSlidesXML	Specifies the input parameters of the **GetSlidesXML** operation.
GetSlidesXMLResponse	Specifies the output of the **GetSlidesXML** operation.

3.1.4.2.2.1 GetSlidesXML

The **GetSlidesXML** element specifies the input parameters of the **GetSlidesXML** operation.

```
<s:element name="GetSlidesXML">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
```

strListUrl: This element MUST be present, as specified in section 2.2.3.1.

3.1.4.2.2.2 GetSlidesXMLResponse

The **GetSlidesXMLResponse** element specifies the output of the **GetSlidesXML** operation.

```
<s:element name="GetSlidesXMLResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="GetSlidesXMLResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
               type="tns:CT_Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetSlidesXMLResult: This element specifies the result of the **GetSlidesXML** operation.

GetSlidesXMLResult.results: This element specifies the result of the **GetSlidesXML** operation.

If **strListUrl** is not present in the **GetSlidesXMLSoapIn** message or is not a folder of a Slide Library, the protocol server MUST return a fault.

Otherwise, the protocol server MUST send a **GetSlidesXMLSoapOut** message, including a single **results** element with a **Slide** element for each of the slides in the folder of the slide library.

3.1.4.3 GetSlideInfoByIds

The **GetSlideInfoByIds** operation is used to get information about slides specified by their identifiers in a folder of a Slide Library.

```
<wsdl:operation name="GetSlideInfoByIds">
      <wsdl:input message="tns:GetSlideInfoByIdsSoapIn" />
      <wsdl:output message="tns:GetSlideInfoByIdsSoapOut" />
</wsdl:operation>
```

The protocol client sends a **GetSlideInfoByIdsSoapIn** request message and the protocol server responds with a **GetSlideInfoByIdsSoapOut** response message.

3.1.4.3.1 Messages

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

Message	Description
GetSlideInfoByIdsSoapIn	The request message for getting information about specific slides in a Slide Library.
GetSlideInfoByIdsSoapOut	The response message to a **GetSlideInfoByIdsSoapIn** request message.

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

3.1.4.3.1.1 GetSlideInfoByIdsSoapIn

This is the request message for getting information about specific slides in a Slide Library.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlideInfoByIds
```

The SOAP body MUST contain a **GetSlideInfoByIds** element.

3.1.4.3.1.2 GetSlideInfoByIdsSoapOut

This is the response message to a **GetSlideInfoByIdsSoapIn** request message.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlideInfoByIds
```

The SOAP body MUST contain a **GetSlideInfoByIdsResponse** element.

3.1.4.3.2 Elements

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

Element	Description
GetSlideInfoByIds	Specifies the input parameters of a **GetSlideInfoByIds** operation.
GetSlideInfoByIdsResponse	Specifies the output of a **GetSlideInfoByIds** operation.

3.1.4.3.2.1 GetSlideInfoByIds

The **GetSlideInfoByIds** element specifies the input parameters of a **GetSlideInfoByIds** operation.

```
<s:element name="GetSlideInfoByIds">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="slideids"
                type="tns:ArrayOfString" />
    </s:sequence>
  </s:complexType>
</s:element>
```

strListUrl: This element MUST be present, as specified in section 2.2.3.1.

slideids: List of identifiers of the slides to be checked.

3.1.4.3.2.2 GetSlideInfoByIdsResponse

The **GetSlideInfoByIdsResponse** element specifies the output of a **GetSlideInfoByIds** operation.

```
<s:element name="GetSlideInfoByIdsResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="GetSlideInfoByIdsResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
               type="tns:CT_Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetSlideInfoByIdsResult: This element specifies the result of the **GetSlideInfoByIds** operation.

GetSlideInfoByIdsResult.results: This element specifies the result of the **GetSlideInfoByIds** operation.

The protocol server MUST send a **GetSlideInfoByIdsSoapOut** message.

If a **string** in the **slideids** list matches a string representation of the identifier of a slide in a folder of the Slide Library, the **GetSlideInfoByIdsSoapOut** message MUST include a single **results** element with a **Slide** element for each of the slides.

3.1.4.4 Search

This operation is used to search for slides that contain the specified search string in a folder of Slide Library.

```
<wsdl:operation name="Search">
      <wsdl:input message="tns:SearchSoapIn" />
      <wsdl:output message="tns:SearchSoapOut" />
</wsdl:operation>
```

The protocol client sends a **SearchSoapIn** request message and the protocol server responds with a **SearchSoapOut** response message.

3.1.4.4.1 Messages

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

Message	Description
SearchSoapIn	The request message for getting information about slides that contain the specified search string in the Slide Library.
SearchSoapOut	The response message to a **SearchSoapIn** request message

3.1.4.4.1.1 SearchSoapIn

This is the request message for getting information about slides that contain the specified search string in the Slide Library.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/Search
```

The SOAP body MUST contain a **Search** element.

3.1.4.4.1.2 SearchSoapOut

This is the response message to a **SearchSoapIn** request message.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/Search
```

The SOAP body MUST contain a **SearchResponse** element.

3.1.4.4.2 Elements

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

Element	Description
Search	Specifies the input parameters for a **Search** operation.
SearchResponse	Specifies the output of a **Search** operation.

3.1.4.4.2.1 Search

The **Search** element specifies the input parameters for a **Search** operation.

```
<s:element name="Search">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="strSearch" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="inputLcid"
                 type="s:unsignedInt" />
    </s:sequence>
  </s:complexType>
</s:element>
```

strListUrl: This element, as specified in section 2.2.3.1, MUST be present and MUST be the URL of a folder of a Slide Library.

strSearch: The string to search for in the slide.

inputLcid: The **language code identifier (LCID)** of **strSearch**. The **inputLcid** MUST be present in a **SearchSoapIn** message and MUST be a valid LCID.

3.1.4.4.2.2 SearchResponse

The **SearchResponse** element specifies the output for a **Search** operation.

```
<s:element name="SearchResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="SearchResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
               type="tns:CT_Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

SearchResult: This element specifies the result of the **Search** operation.

SearchResult.results: This element specifies the result of the **Search** operation.

If the search operation returns a fault, the server MUST return a SOAP fault with the error code set to the following value:

0x00000008: The search operation cannot be finished successfully.

Otherwise, the server MUST send a **SearchSoapOut** message.

If a slide contains **strSearch**, the **SearchSoapOut** message MUST include a **results** element with a **Slide** element for each matching slide.

The **results** element of the **SearchSoapOut** message MUST NOT contain more than 50 **Slide** elements.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

The following examples demonstrate the interactions between the protocol client and the protocol server. For the sake of succinctness, only the SOAP body is listed.

In this example, the protocol client searches for slides that contain content about the word "Zune". The **SearchSoapIn** message is sent to the protocol server:

```
<Search xmlns="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/">
  <strListUrl>http://sharepoint/sites/ddk/First Slide Lib/</strListUrl>
  <strSearch>Zune</strSearch>
  <inputLcid>1033</inputLcid>
</Search>
```

In this example, the protocol server searches the slide library "http://sharepoint/sites/ddk/First Slide Lib/" for slides that contain the English (LCID 1033) word "Zune" and returns the result that describes the two slides found:

```
<SearchResponse>
<SearchResult>
  <results xmlns="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/">
    <Title>First Slide Lib</Title>
  <Url>http://sharepoint/sites/ddk/First%20Slide%20Lib</Url>
    <ForceCheckout>False</ForceCheckout>
    <Slide FileName="Zune Launch Overview 9-22 272.pptx" ID="56" UniqueId="ecc2f5ff-e0a8-
445f-b254-d1be6a39cc8e" Presentation="Zune Launch Overview 9-22"
EncodedAbsUrl="http://sharepoint/sites/ddk/First%20Slide%20Lib/Zune%20Launch%20Overview%209-
22_272.pptx"
EncodedAbsThumbnailUrl="http://sharepoint/sites/ddk/First%20Slide%20Lib/_t/Zune%20Launch%20Ov
erview%209-22 272 pptx.jpg" Editor="GPU Test Lab" Description="Seattle: Pike Place Market"
Modified="2006-12-06T19:10:50Z" />
    <Slide FileName="Zune Launch Overview 9-22 220.pptx" ID="75" UniqueId="08e3bf1e-880c-
4e09-be8b-a93452e5059d" Presentation="Zune Launch Overview 9-22"
EncodedAbsUrl="http://sharepoint/sites/ddk/First%20Slide%20Lib/Zune%20Launch%20Overview%209-
22_220.pptx"
EncodedAbsThumbnailUrl="http://sharepoint/sites/ddk/First%20Slide%20Lib/ t/Zune%20Launch%20Ov
erview%209-22_220_pptx.jpg" Editor="GPU Test Lab" Description="Flash Performance Promotion -
Sub Market" Modified="2006-12-06T19:11:04Z" />
  </results>
</SearchResult>
</SearchResponse>
```

The protocol client can download the content of these two slides using URLs included as the **EncodedAbsUrl** attribute in each **Slide** element, or the thumbnail images of these two slides using URLs included as the **EncodedAbsThumbnailUrl** attribute.

5 Security

5.1 Security Considerations for Implementers

Security assumptions of this protocol are documented in [MS-AUTHWS], section 5. There are no additional security considerations for implementers.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL are provided in this appendix:

```
<?xml version="1.0" encoding="utf-8" ?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/">
      <s:import namespace="http://microsoft.com/wsdl/types/" />
      <s:import namespace="http://www.w3.org/2001/XMLSchema" />
      <s:complexType name="CT SlideResult">
        <s:attribute use="optional" name="FileName" type="s:string" />
        <s:attribute use="optional" name="Description" type="s:string" />
        <s:attribute use="optional" name="Editor" type="s:string" />
        <s:attribute use="optional" name="EncodedAbsThumbnailUrl"
               type="s:string" />
        <s:attribute use="required" name="Modified" type="s:string" />
        <s:attribute use="required" name="ID" type="s:unsignedInt" />
        <s:attribute use="required" name="EncodedAbsUrl" type="s:string" />
        <s:attribute use="optional" name="Presentation" type="s:string" />
        <s:attribute use="optional" name="UniqueId" type="s:string" />
      </s:complexType>
      <s:simpleType name="ST TrueFalse">
        <s:restriction base="s:string">
          <s:enumeration value="true"/>
          <s:enumeration value="false"/>
          <s:enumeration value="True"/>
          <s:enumeration value="False"/>
        </s:restriction>
      </s:simpleType>
      <s:complexType name="CT_Result">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="Title" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Url" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="ForceCheckout"
               type="tns:ST_TrueFalse" />
          <s:element minOccurs="0" maxOccurs="unbounded" name="Slide"
               type="tns:CT_SlideResult" />
        </s:sequence>
      </s:complexType>
      <s:element name="GetSlidesXML">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="strListUrl"
               type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetSlidesXMLResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="GetSlidesXMLResult">
              <s:complexType>
                <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="results"
                             type="tns:CT_Result" />
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
```

```
</s:element>
<s:element name="CheckCollisions">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl"
         type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="slideNames"
         type="tns:ArrayOfString" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string"
         nillable="true" type="s:string" />
  </s:sequence>
</s:complexType>
<s:element name="CheckCollisionsResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="CheckCollisionsResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
         type="tns:CT_Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetSlideInfoByIds">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl"
         type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="slideids"
         type="tns:ArrayOfString" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetSlideInfoByIdsResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="GetSlideInfoByIdsResult">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="results"
                       type="tns:CT_Result" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="Search">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="strListUrl"
                 type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="strSearch"
                 type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="inputLcid"
                 type="s:unsignedInt" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="SearchResponse">
  <s:complexType>
```

```
            <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="SearchResult">
                <s:complexType>
                  <s:sequence>
                    <s:element minOccurs="1" maxOccurs="1" name="results"
                               type="tns:CT Result" />
                  </s:sequence>
                </s:complexType>
              </s:element>
            </s:sequence>
          </s:complexType>
        </s:element>
      </s:schema>
      <s:schema elementFormDefault="qualified"
                      targetNamespace="http://microsoft.com/wsdl/types/">
        <s:simpleType name="guid">
          <s:restriction base="s:string">
            <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-
9a-fA-F]{12}" />
          </s:restriction>
        </s:simpleType>
      </s:schema>
    </wsdl:types>
    <wsdl:message name="GetSlidesXMLSoapIn">
      <wsdl:part name="parameters" element="tns:GetSlidesXML" />
    </wsdl:message>
    <wsdl:message name="GetSlidesXMLSoapOut">
      <wsdl:part name="parameters" element="tns:GetSlidesXMLResponse" />
    </wsdl:message>
    <wsdl:message name="CheckCollisionsSoapIn">
      <wsdl:part name="parameters" element="tns:CheckCollisions" />
    </wsdl:message>
    <wsdl:message name="CheckCollisionsSoapOut">
      <wsdl:part name="parameters" element="tns:CheckCollisionsResponse" />
    </wsdl:message>
    <wsdl:message name="GetSlideInfoByIdsSoapIn">
      <wsdl:part name="parameters" element="tns:GetSlideInfoByIds" />
    </wsdl:message>
    <wsdl:message name="GetSlideInfoByIdsSoapOut">
      <wsdl:part name="parameters" element="tns:GetSlideInfoByIdsResponse" />
    </wsdl:message>
    <wsdl:message name="SearchSoapIn">
      <wsdl:part name="parameters" element="tns:Search" />
    </wsdl:message>
    <wsdl:message name="SearchSoapOut">
      <wsdl:part name="parameters" element="tns:SearchResponse" />
    </wsdl:message>
    <wsdl:portType name="SlideLibrarySoap">
      <wsdl:operation name="GetSlidesXML">
        <wsdl:input message="tns:GetSlidesXMLSoapIn" />
        <wsdl:output message="tns:GetSlidesXMLSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="CheckCollisions">
        <wsdl:input message="tns:CheckCollisionsSoapIn" />
        <wsdl:output message="tns:CheckCollisionsSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetSlideInfoByIds">
        <wsdl:input message="tns:GetSlideInfoByIdsSoapIn" />
        <wsdl:output message="tns:GetSlideInfoByIdsSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="Search">
        <wsdl:input message="tns:SearchSoapIn" />
        <wsdl:output message="tns:SearchSoapOut" />
      </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="SlideLibrarySoap" type="tns:SlideLibrarySoap">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document" />
      <wsdl:operation name="GetSlidesXML">
```

```
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlidesXML"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CheckCollisions">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/CheckCollisions"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSlideInfoByIds">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlideInfoByIds"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Search">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/Search"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="SlideLibrarySoap12" type="tns:SlideLibrarySoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" style="document" />
    <wsdl:operation name="GetSlidesXML">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlidesXML"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CheckCollisions">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/CheckCollisions"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetSlideInfoByIds">
```

```
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/GetSlideInfoByIds"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Search">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/SlideLibrary/Search"
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 PowerPoint 2007
- Microsoft PowerPoint 2010
- Microsoft Office SharePoint Server 2007
- Microsoft SharePoint Server 2010
- Microsoft PowerPoint 2013
- Microsoft SharePoint Server 2013
- Microsoft PowerPoint 2016
- Microsoft SharePoint Server 2016

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 this document since the last release. 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.
- The removal of a document from the documentation set.

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 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 changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

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.
- 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 dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

9 Index

T

V

W