

[MS-PWBHPS]: PowerPoint Web Broadcast Host 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 [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.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
07/13/2009	0.1	Major	Initial Availability
08/28/2009	0.2	Editorial	Revised and edited the technical content
11/06/2009	0.3	Editorial	Revised and edited the technical content
02/19/2010	1.0	Major	Updated and revised the technical content
03/31/2010	1.01	Editorial	Revised and edited the technical content
04/30/2010	1.02	Editorial	Revised and edited the technical content
06/07/2010	1.03	Editorial	Revised and edited the technical content
06/29/2010	1.04	Minor	Clarified the meaning of the technical content.
07/23/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	1.05	Minor	Clarified the meaning of the technical content.
11/15/2010	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.06	Editorial	Changed language and formatting in the technical content.
03/18/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	1.06	No change	No changes to the meaning, language, or formatting 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	8
1.6 Applicability Statement	8
1.7 Versioning and Capability Negotiation	8
1.8 Vendor-Extensible Fields	8
1.9 Standards Assignments	8
2 Messages	9
2.1 Transport	9
2.2 Common Message Syntax	9
2.2.1 Namespaces	9
2.2.2 Messages	9
2.2.3 Elements	9
2.2.4 Complex Types	10
2.2.4.1 ArrayOfCapabilityData	10
2.2.4.2 BroadcastFile	10
2.2.4.3 CapabilityData	11
2.2.4.4 ServerInfo	12
2.2.4.5 ServiceError	12
2.2.4.6 ServiceResult	13
2.2.4.7 Version	13
2.2.5 Simple Types	14
2.2.5.1 ClientActions	14
2.2.5.2 ServerCapability	15
2.2.5.3 ServiceErrorType	15
2.2.6 Attributes	16
2.2.7 Groups	16
2.2.8 Attribute Groups	16
3 Protocol Details	17
3.1 Server Details	17
3.1.1 Abstract Data Model	19
3.1.2 Timers	19
3.1.3 Initialization	20
3.1.4 Message Processing Events and Sequencing Rules	20
3.1.4.1 BroadcastDeleteUploadFile	20
3.1.4.1.1 Messages	20
3.1.4.1.1.1 BroadcastDeleteUploadFileSoapIn	20
3.1.4.1.1.2 BroadcastDeleteUploadFileSoapOut	21
3.1.4.1.2 Elements	21
3.1.4.1.2.1 BroadcastDeleteUploadFile	21
3.1.4.1.2.2 BroadcastDeleteUploadFileResponse	21
3.1.4.1.3 Complex Types	21
3.1.4.1.4 Simple Types	22

3.1.4.1.5	Attributes.....	22
3.1.4.1.6	Groups.....	22
3.1.4.1.7	Attribute Groups	22
3.1.4.2	BroadcastGetAttendeeUrl.....	22
3.1.4.2.1	Messages	22
3.1.4.2.1.1	BroadcastGetAttendeeUrlSoapIn.....	22
3.1.4.2.1.2	BroadcastGetAttendeeUrlSoapOut.....	22
3.1.4.2.2	Elements.....	22
3.1.4.2.2.1	BroadcastGetAttendeeUrl.....	23
3.1.4.2.2.2	BroadcastGetAttendeeUrlResponse	23
3.1.4.2.3	Complex Types	23
3.1.4.2.3.1	BroadcastUser.....	23
3.1.4.2.4	Simple Types.....	24
3.1.4.2.5	Attributes.....	24
3.1.4.2.6	Groups.....	24
3.1.4.2.7	Attribute Groups	24
3.1.4.3	BroadcastGetHostToken	24
3.1.4.3.1	Messages	24
3.1.4.3.1.1	BroadcastGetHostTokenSoapIn	24
3.1.4.3.1.2	BroadcastGetHostTokenSoapOut	24
3.1.4.3.2	Elements.....	25
3.1.4.3.2.1	BroadcastGetHostToken	25
3.1.4.3.2.2	BroadcastGetHostTokenResponse	25
3.1.4.3.3	Complex Types	25
3.1.4.3.4	Simple Types.....	25
3.1.4.3.5	Attributes.....	25
3.1.4.3.6	Groups.....	25
3.1.4.3.7	Attribute Groups	26
3.1.4.4	BroadcastGetNewUploadFile.....	26
3.1.4.4.1	Messages	26
3.1.4.4.1.1	BroadcastGetNewUploadFileSoapIn	26
3.1.4.4.1.2	BroadcastGetNewUploadFileSoapOut.....	26
3.1.4.4.2	Elements.....	26
3.1.4.4.2.1	BroadcastGetNewUploadFile.....	26
3.1.4.4.2.2	BroadcastGetNewUploadFileResponse	27
3.1.4.4.3	Complex Types	27
3.1.4.4.4	Simple Types.....	27
3.1.4.4.5	Attributes.....	27
3.1.4.4.6	Groups.....	27
3.1.4.4.7	Attribute Groups	27
3.1.4.5	BroadcastGetServerInfo	27
3.1.4.5.1	Messages	27
3.1.4.5.1.1	BroadcastGetServerInfoSoapIn	28
3.1.4.5.1.2	BroadcastGetServerInfoSoapOut	28
3.1.4.5.2	Elements.....	28
3.1.4.5.2.1	BroadcastGetServerInfo	28
3.1.4.5.2.2	BroadcastGetServerInfoResponse	28
3.1.4.5.3	Complex Types	29
3.1.4.5.3.1	ClientInfo	29
3.1.4.5.4	Simple Types.....	29
3.1.4.5.5	Attributes.....	29
3.1.4.5.6	Groups.....	29
3.1.4.5.7	Attribute Groups	29

3.1.5	Timer Events	29
3.1.6	Other Local Events	29
3.2	Client Details.....	29
3.2.1	Abstract Data Model	29
3.2.2	Timers	30
3.2.3	Initialization	30
3.2.4	Message Processing Events and Sequencing Rules.....	30
3.2.5	Timer Events	30
3.2.6	Other Local Events	30
4	Protocol Examples.....	31
4.1	Retrieving Protocol Server Information	31
4.2	Obtaining the Upload URL.....	32
4.3	Uploading the Presentation to the Upload URL.....	32
4.4	Starting the Broadcast Session	33
4.5	Retrieving the Attendee URL for the Broadcast	33
4.6	Using the BroadcastPutDataSoapIn Message	34
4.7	Ending the Broadcast Session.....	34
4.8	Deleting the Uploaded Presentation.....	35
5	Security.....	36
5.1	Security Considerations for Implementers.....	36
5.2	Index of Security Parameters	36
6	Appendix A: Full WSDL.....	37
7	Appendix B: Product Behavior	44
8	Change Tracking.....	45
9	Index	46

1 Introduction

This document specifies the PowerPoint Web Broadcast Host Protocol, which enables a protocol client to upload a **presentation** to a protocol server, delete a presentation from a protocol server, and retrieve information that is necessary to successfully initiate a **slide show broadcast** session.

Sections 1.8, 2, and 3 of this specification are normative and contain RFC 2119 language. Sections 1.5 and 1.9 are also normative but cannot contain RFC 2119 language. All other sections and examples in this specification are informative.

1.1 Glossary

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

Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)

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

handout master slide
presentation
Simple Object Access Protocol (SOAP)
slide show broadcast
SOAP action
SOAP body
SOAP fault
Uniform Resource Locator (URL)
Web Distributed Authoring and Versioning Protocol (WebDAV)
Web Services Description Language (WSDL)
Web site
WSDL message
WSDL operation
XML namespace

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

References to Microsoft Open Specification documents do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

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-PWBPS] Microsoft Corporation, "[PowerPoint Web Broadcast 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>
- [RFC2518] Goland, Y., Whitehead, E., Faizi, A., et al., "HTTP Extensions for Distributed Authoring - WebDAV", RFC 2518, February 1999, <http://www.ietf.org/rfc/rfc2518.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>
- [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 enables a protocol client to upload a presentation to a protocol server, delete a presentation from a protocol server, and retrieve information that is necessary to successfully initiate a slide show broadcast session.

The protocol client needs to interact with the protocol server and the PowerPoint Web Broadcast service as specified in [\[MS-PWBPS\]](#) to initiate a slide show broadcast session.

The protocol allows the protocol client to request information such as the protocol server settings, **URL** to upload the presentation to, location of the PowerPoint Web Broadcast service and the URL to view the slide show broadcast session.

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 this protocol:

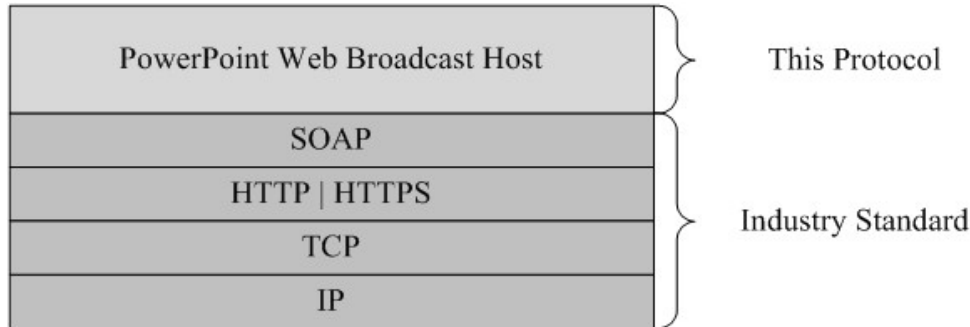


Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a **Web site (1)** that is identified by a URL that is known by protocol clients. The protocol server endpoint is formed by appending `"/_vti_bin/PowerPointBroadcastHost_1_0.asmx"` to the URL of the Web site, for example `http://www.contoso.com/sites/broadcast/_vti_bin/PowerPointBroadcastHost_1_0.asmx`.

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

1.6 Applicability Statement

This protocol is designed to upload a presentation to a protocol server, delete a presentation from a protocol server, and retrieve information that is necessary to successfully begin a slide show broadcast session.

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**, **present**, and **not null**.

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 in either [\[SOAP1.1\]](#) section 4.4 or in [\[SOAP1.2/1\]](#) section 5.4.

2.2 Common Message Syntax

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

2.2.1 Namespaces

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

Prefix	Namespace URI	Reference
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
tns	http://schemas.microsoft.com/server/powerpoint/2009/main	
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]

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 operation are described with the operation.

Complex Type	Description
ArrayOfCapabilityData	A complex type that specifies a list of CapabilityData elements.
BroadcastFile	A complex type that specifies the upload URL on the protocol server and identifier of a presentation.
CapabilityData	A complex type that specifies an individual capability returned by the protocol server to a protocol client.
ServerInfo	The ServerInfo type specifies the server information returned by the protocol server to a protocol client.
ServiceError	A complex type that specifies error information returned by the protocol server to a protocol client.
ServiceResult	A complex type that specifies the result of an operation. The protocol server returns this type to the protocol client containing either a successful Result element or an Error element.
Version	A complex type that specifies a version.

2.2.4.1 ArrayOfCapabilityData

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

A complex type that specifies a list of [CapabilityData](#) elements.

```
<xs:complexType name="ArrayOfCapabilityData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="CapabilityData" nillable="true"
      type="tns:CapabilityData"/>
  </xs:sequence>
</xs:complexType>
```

CapabilityData: Each element MUST specify a [CapabilityData](#).

2.2.4.2 BroadcastFile

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

A complex type that specifies the upload URL on the protocol server and identifier of a presentation.

```
<xs:complexType name="BroadcastFile">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="FilePath" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="PresentationId" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

FilePath: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) specifies the upload URL of the presentation on the protocol server.

PresentationId: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identifier of the presentation.

2.2.4.3 CapabilityData

Namespace: http://schemas.microsoft.com/server/powerpoint/2009/main

A complex type that specifies an individual capability returned by the protocol server to a protocol client.

```
<xs:complexType name="CapabilityData">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="Capability" type="tns:ServerCapability"/>
    <xs:element minOccurs="0" maxOccurs="1" name="Value" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Capability: A [ServerCapability](#) element that specifies the capability.

Value: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the value for that capability. **Value** MUST be as specified in the following table based on the value of **Capability**.

Capability	Value
SupportAudio	The Value element MUST be "true" or "false". If "false", the protocol client MUST remove all audio data from the presentation before uploading it to the protocol server. If "true", the protocol client MAY upload a presentation containing audio data. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of "false" for this Capability.
SupportVideo	The Value element MUST be "true" or "false". If "false", the protocol client MUST remove all video data from the presentation before uploading it to the protocol server. If "true", the protocol client MAY upload a presentation containing video data. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of "false" for this Capability.
SupportNotes	The Value element MUST be "true" or "false". If "false", the protocol client MUST remove all notes data from the presentation before uploading it to the protocol server. If "true", the protocol client MAY upload a presentation containing notes data. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of "false" for this Capability.
SupportHandoutMaster	The Value element MUST be "true" or "false". If "false", the protocol client MUST remove all handout master slide data from the presentation before uploading it to the protocol server. If "true", the protocol client MAY upload a presentation containing handout master slide data. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of "false" for this Capability.
FileSizeLimited	The Value element MUST be greater than or equal to zero. The Value element MUST specify the protocol's maximum presentation file size in bytes. If the Value element is zero, this MUST specify that the protocol server does not limit the presentation file size. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of zero for this Capability.

Capability	Value
UseWebDav	The Value element MUST be "true" or "false". If the Value element is "true", the protocol client MUST use WebDAV to upload the presentation to the protocol server. Otherwise the protocol client MUST NOT use WebDAV. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of "false" for this Capability.
SessionTimeout	The Value element MUST be greater than or equal to zero. The Value element MUST specify the maximum duration for a broadcast session after which the client MUST end the session. If the Value element is zero, this MUST specify that the protocol server does not limit the session duration. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of zero for this Capability.
SessionIdleTimeout	The Value element MUST be greater than or equal to zero. The Value element MUST specify the maximum duration a broadcast session can be idle, after which the client MUST end the broadcast session. If the Value element is zero, this MUST specify that the protocol server does not limit the session idle duration. If this Capability is not sent by the protocol server, the protocol client MUST use the default value of zero for this Capability.

2.2.4.4 ServerInfo

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

The **ServerInfo** type specifies the server information returned by the protocol server to a protocol client.

```
<xs:complexType name="ServerInfo">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="Name" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="Version" type="tns:Version"/>
    <xs:element minOccurs="0" maxOccurs="1" name="PresenterServiceUrl" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="CapabilitiesList"
      type="tns:ArrayOfCapabilityData"/>
  </xs:sequence>
</xs:complexType>
```

Name: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the protocol server name.

Version: A [Version](#) element that specifies the protocol server version.

PresenterServiceUrl: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the URL of the presenter service.

CapabilitiesList: An [ArrayOfCapabilityData](#) element that specifies the capabilities supported by the protocol server.

2.2.4.5 ServiceError

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

A complex type that specifies error information returned by the protocol server to a protocol client.

```
<xs:complexType name="ServiceError">
  <xs:sequence>
```

```

<xs:element minOccurs="0" maxOccurs="1" name="Message" type="xs:string"/>
<xs:element minOccurs="0" maxOccurs="1" name="Title" type="xs:string"/>
<xs:element minOccurs="1" maxOccurs="1" name="Type" type="tns:ServiceErrorType"/>
<xs:element minOccurs="1" maxOccurs="1" name="RecommendedActions"
type="tns:ClientActions"/>
</xs:sequence>
</xs:complexType>

```

Message: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the error message description. The string length **MUST** be greater than zero if the **Type** element has a value of [ApplicationError](#). This element **MUST** be present.

Title: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the error title. The string length **MUST** be greater than zero if the **Type** element has a value of [ApplicationError](#). This element **MUST** be present.

Type: A [ServiceErrorType](#) element that specifies the error type. This element **MUST** be present.

RecommendedActions: Reserved. **MUST** be ignored.

2.2.4.6 ServiceResult

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

A complex type that specifies the result of an operation. The protocol server returns this type to the protocol client containing either a successful **Result** element or an **Error** element.

```

<xs:complexType name="ServiceResult">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="Result"/>
    <xs:element minOccurs="0" maxOccurs="1" name="Error" type="tns:ServiceError"/>
  </xs:sequence>
</xs:complexType>

```

Result: An optional **s:anyType** element ([\[XMLSCHEMA1\]](#) section 3.4.7) that specifies a successful result of a protocol message response. This element **MUST NOT** be present if the **Error** element is present.

Error: An optional [ServiceError](#) element that specifies an error result of a protocol message response. This element **MUST NOT** be present if the **Result** element is present.

2.2.4.7 Version

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

A complex type that specifies a version.

```

<xs:complexType name="Version">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="MajorNumber" type="xs:int"/>
    <xs:element minOccurs="1" maxOccurs="1" name="MinorNumber" type="xs:int"/>
  </xs:sequence>
</xs:complexType>

```

MajorNumber: An **s:int** element ([\[XMLSCHEMA2\]](#) section 3.3.13) that specifies the major part of the version.

MinorNumber: An **s:int** element ([\[XMLSCHEMA2\]](#) section 3.3.13) that specifies the minor part of the version.

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
ClientActions	Reserved. MUST be ignored.
ServerCapability	A simple type that specifies an enumeration of a set of capabilities returned by the protocol server to the protocol client.
ServiceErrorType	A simple type that specifies an enumeration of a set of protocol errors returned by the protocol server to the protocol client.

2.2.5.1 ClientActions

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

Reserved. MUST be ignored.

```
<xs:simpleType name="ClientActions">
  <xs:list>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="None"/>
        <xs:enumeration value="Dismiss"/>
        <xs:enumeration value="Close"/>
        <xs:enumeration value="OpenInClient"/>
        <xs:enumeration value="Refresh"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:list>
</xs:simpleType>
```

The following table specifies the allowable values for ClientActions:

Value	Meaning
None	Reserved. MUST be ignored.
Dismiss	Reserved. MUST be ignored.
Close	Reserved. MUST be ignored.
OpenInClient	Reserved. MUST be ignored.
Refresh	Reserved. MUST be ignored.

2.2.5.2 ServerCapability

Namespace: http://schemas.microsoft.com/server/powerpoint/2009/main

A simple type that specifies an enumeration of a set of capabilities returned by the protocol server to the protocol client.

```
<xs:simpleType name="ServerCapability">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SupportAudio"/>
    <xs:enumeration value="SupportVideo"/>
    <xs:enumeration value="SupportNotes"/>
    <xs:enumeration value="SupportHandoutMaster"/>
    <xs:enumeration value="FileSizeLimited"/>
    <xs:enumeration value="UseWebDav"/>
    <xs:enumeration value="SessionTimeout"/>
    <xs:enumeration value="SessionIdleTimeout"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for ServerCapability:

Value	Meaning
SupportAudio	This capability specifies whether the protocol server supports audio content in the presentation.
SupportVideo	This capability specifies whether the protocol server supports video content in the presentation.
SupportNotes	This capability specifies whether the protocol server supports notes in the presentation.
SupportHandoutMaster	This capability specifies whether the protocol server supports handout master slide in the presentation.
FileSizeLimited	This capability specifies what the protocol server requirements are for maximum presentation file size.
UseWebDav	This capability specifies whether the protocol server supports the WebDAV protocol for uploading files to a server.
SessionTimeout	This capability specifies the maximum duration for a broadcast session after which the client MUST end the session.
SessionIdleTimeout	This capability specifies the maximum duration a broadcast session can be idle for after which the client MUST end the broadcast session.

2.2.5.3 ServiceErrorType

Namespace: http://schemas.microsoft.com/server/powerpoint/2009/main

A simple type that specifies an enumeration of a set of protocol errors returned by the protocol server to the protocol client.

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

```
<xs:enumeration value="UnknownError"/>
<xs:enumeration value="ApplicationError"/>
<xs:enumeration value="Timeout"/>
</xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for ServiceErrorType:

Value	Meaning
UnknownError	The protocol server encountered an unknown error.
ApplicationError	The protocol server encountered an application error.
Timeout	The protocol server encountered an application timeout.

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**, **present**, and **not null**.

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 allows protocol servers to notify protocol clients of application-level faults using SOAP faults. Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

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

3.1 Server Details

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

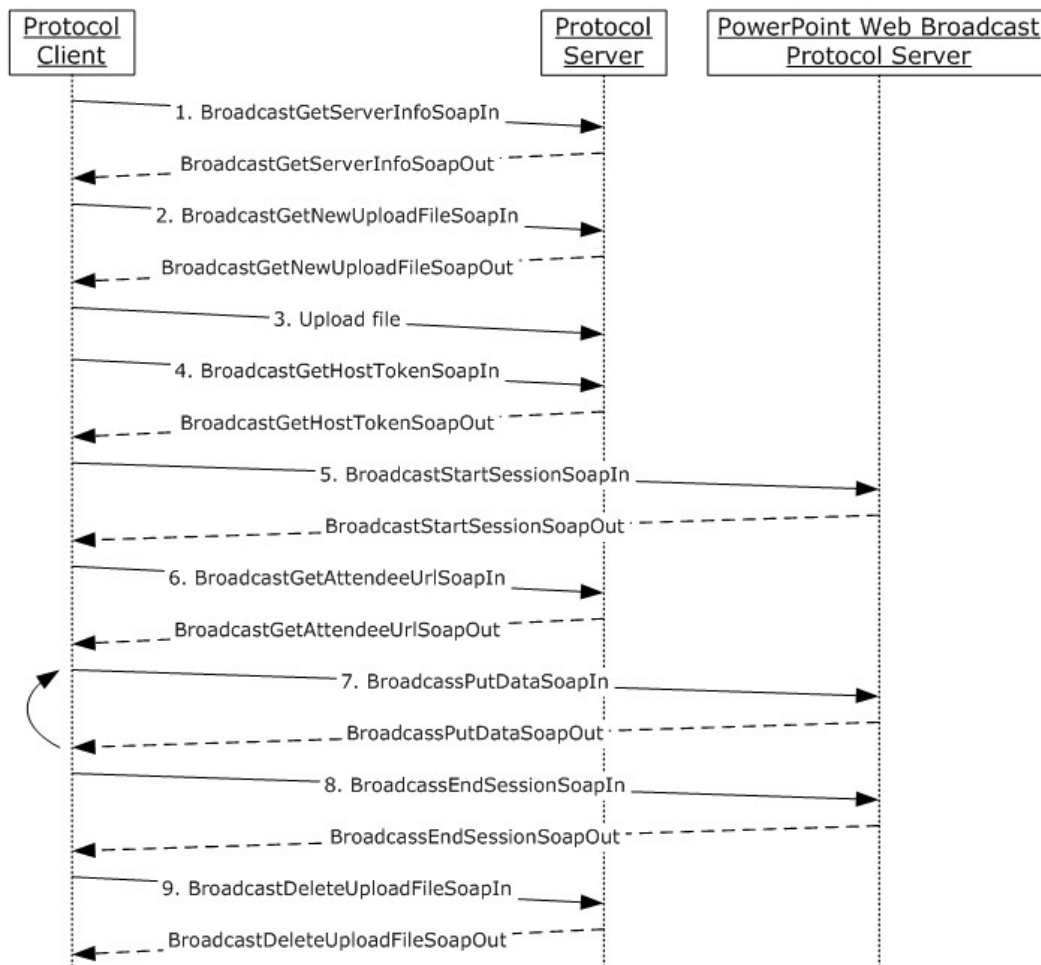


Figure 2: Communication between protocol client and protocol server

The protocol client sends a **BroadcastGetServerInfoSoapIn** message (section 3.1.4.1.1.1) to the protocol server to retrieve the protocol server information. The protocol server sends [ServerInfo](#) element to the protocol client which contains a PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server endpoint along with a list of capabilities supported by the protocol server in the **BroadcastGetServerInfoSoapOut** message (section 3.1.4.1.1.2).

The protocol client MUST honor the capability information sent by the protocol server. For example, if the protocol server sends [SupportAudio](#) capability as "false", then the protocol client MUST remove all audio data from the presentation before uploading it to the protocol server. The protocol client MUST make sure that the presentation size is less than the size specified by the protocol server in `FileSizeLimited` capability.

The protocol client then sends a **BroadcastGetNewUploadFileSoapIn** message to the protocol server to get an upload URL for the presentation. The protocol server returns the upload URL in the **BroadcastGetNewUploadFileSoapOut** message.

If the `UseWebDav` capability sent by the protocol server in **BroadcastGetServerInfoSoapOut** message is set to "true", the protocol client then uploads the presentation to the upload URL using the WebDAV PUT request as specified in [\[RFC2518\]](#). Otherwise, the protocol client sends an HTTP

POST request to the upload URL with the presentation data included in the HTTP POST message body.

Once a presentation is uploaded by the protocol client, it sends a **BroadcastGetHostTokenSoapIn** message to the protocol server to get a host token, an **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1), corresponding to the uploaded file. The protocol server returns the host token in the corresponding **BroadcastGetHostTokenSoapOut** message.

The protocol client then starts the slide show broadcast session by sending a **BroadcastStartSessionSoapIn** message to the PowerPoint Web Broadcast Protocol [MS-PWBPS] server. The protocol client sends the host token retrieved from the protocol server in the **BroadcastStartSessionSoapIn** message. PowerPoint Web Broadcast Protocol [MS-PWBPS] server returns a **BroadcastUser** [\[MS-PWBPS\]](#) section 2.2.4.1 element to the protocol client in **BroadcastStartSessionSoapOut** message.

The protocol client sends the **BroadcastUser** [\[MS-PWBPS\]](#) section 2.2.4.1 element retrieved from the **BroadcastStartSessionSoapOut** message to the protocol server in **BroadcastGetAttendeeUrlSoapIn** message. The protocol server forms an attendee URL and sends it to the protocol client in the **BroadcastGetAttendeeUrlSoapOut** message. The attendee URL is then distributed to the slide show broadcast attendees by the presenter who is the owner of the presentation that is being requested.

The protocol client sends **BroadcastPutDataSoapIn** messages to the PowerPoint Web Broadcast Protocol [MS-PWBPS] server for the duration of the slide show broadcast session. At the end of the slide show broadcast session the protocol client sends **BroadcastEndSessionSoapIn** message to the PowerPoint Web Broadcast Protocol [MS-PWBPS] server to end the session.

The protocol client finally sends **BroadcastDeleteUploadedFileSoapIn** message to the protocol server. The protocol server deletes the uploaded presentation and responds with a **BroadcastDeleteUploadedFileSoapOut** message.

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.

presentationId: An entity which represents a unique identifier for a presentation.

hostToken: An entity that specifies the token returned by the protocol server corresponding to the presentation that is uploaded by the protocol client.

The protocol server MUST assign a **presentationId** to the presentations that are uploaded to the protocol server. Additionally, the protocol server MUST maintain a host token corresponding to a **presentationId**. The host token is passed to the PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server by the protocol client to associate the corresponding presentation with a slide show broadcast session. PowerPoint Web Broadcast Protocol [MS-PWBPS] server retrieves the presentation from the protocol server by sending the host token for the presentation to the protocol server.

3.1.2 Timers

None.

3.1.3 Initialization

The protocol server MUST expose its Web operations at the following URL, which builds upon a base URL. The URL MUST conform to the following structure: *base URL/_vti_bin/PowerPointBroadcastHost_1_0.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

For information about sequencing of the protocol messages and how they relate to each other, see section [3.1](#). The following sections specify the details of each individual message.

This specification includes the following **WSDL operations**:

WSDL Operation	Description
BroadcastDeleteUploadFile	The BroadcastDeleteUploadFile operation informs the protocol server to delete the specified presentation.
BroadcastGetAttendeeUrl	The BroadcastGetAttendeeUrl operation is used to get the attendee URL of a slide show broadcast from the server.
BroadcastGetHostToken	The BroadcastGetHostToken operation receives a presentation identifier as input and returns a string that represents the host token for that presentation.
BroadcastGetNewUploadFile	The BroadcastGetNewUploadFile operation returns a BroadcastFile (section 2.2.4.2) that contains the upload URL and identifier for the presentation to be used during a slide show broadcast.
BroadcastGetServerInfo	The BroadcastGetServerInfo operation returns information about the protocol server.

3.1.4.1 BroadcastDeleteUploadFile

The **BroadcastDeleteUploadFile** operation informs the protocol server to delete the specified presentation.

```
<wsdl:operation name="BroadcastDeleteUploadFile">
  <wsdl:input message="tns:BroadcastDeleteUploadFileSoapIn"/>
  <wsdl:output message="tns:BroadcastDeleteUploadFileSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastDeleteUploadFileSoapIn** request message and the protocol server MUST respond with a **BroadcastDeleteUploadFileSoapOut** response message as specified in the following sections.

3.1.4.1.1 Messages

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

3.1.4.1.1.1 BroadcastDeleteUploadFileSoapIn

The request WSDL message for the **BroadcastDeleteUploadFile** WSDL operation.

The **SOAP action** value is:

<http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastDeleteUploadFile>

The **SOAP body** contains the **BroadcastDeleteUploadFile** element.

3.1.4.1.1.2 BroadcastDeleteUploadFileSoapOut

The response WSDL message for the **BroadcastDeleteUploadFile** WSDL operation.

The SOAP body contains the **BroadcastDeleteUploadFileResponse** element.

3.1.4.1.2 Elements

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

3.1.4.1.2.1 BroadcastDeleteUploadFile

The input data for the **BroadcastDeleteUploadFile** WSDL operation.

```
<xs:element name="BroadcastDeleteUploadFile">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="presentationId" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

presentationId: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identifier of the presentation to delete. This element **MUST** be present.

3.1.4.1.2.2 BroadcastDeleteUploadFileResponse

The result data for the **BroadcastDeleteUploadFile** WSDL operation.

```
<xs:element name="BroadcastDeleteUploadFileResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastDeleteUploadFileResult"
type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastDeleteUploadFileResult: A [ServiceResult](#) that specifies the result of the operation. This element **MUST** be present. If the **Result** child element is present, it **MUST** be ignored by the protocol client. If the **Error** child element is present, the protocol client **MAY** retry the request or display the error to the user.

3.1.4.1.3 Complex Types

None.

3.1.4.1.4 Simple Types

None.

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.4.2 BroadcastGetAttendeeUrl

The BroadcastGetAttendeeUrl operation is used to get the attendee URL of a slide show broadcast from the server.

```
<wsdl:operation name="BroadcastGetAttendeeUrl">
  <wsdl:input message="tns:BroadcastGetAttendeeUrlSoapIn"/>
  <wsdl:output message="tns:BroadcastGetAttendeeUrlSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastGetAttendeeUrlSoapIn** request message, and the protocol server responds with a **BroadcastGetAttendeeUrlSoapOut** response message, as specified in the following sections.

3.1.4.2.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.2.1.1 BroadcastGetAttendeeUrlSoapIn

The request WSDL message for the **BroadcastGetAttendeeUrl** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetAttendeeUrl
```

The SOAP body contains the **BroadcastGetAttendeeUrl** element.

3.1.4.2.1.2 BroadcastGetAttendeeUrlSoapOut

The response WSDL message for the **BroadcastGetAttendeeUrl** WSDL operation.

The SOAP body contains the **BroadcastGetAttendeeUrlResponse** element.

3.1.4.2.2 Elements

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

3.1.4.2.2.1 BroadcastGetAttendeeUrl

The input data for the **BroadcastGetAttendeeUrl** WSDL operation.

```
<xs:element name="BroadcastGetAttendeeUrl">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="user" type="tns:BroadcastUser"/>
      <xs:element minOccurs="0" maxOccurs="1" name="presentationId" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

user: A [BroadcastUser](#) element that specifies the user which is requesting the attendee URL. The user **MUST** be the owner of the presentation that is being requested.

presentationId: An **s:string** ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identifier of the presentation of the desired attendee URL.

3.1.4.2.2.2 BroadcastGetAttendeeUrlResponse

The result data for the **BroadcastGetAttendeeUrl** WSDL operation.

```
<xs:element name="BroadcastGetAttendeeUrlResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastGetAttendeeUrlResult"
type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastGetAttendeeUrlResult: A [ServiceResult](#) that specifies the result of the operation. If the **Result** child element is present, it **MUST** be an **s:string** ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the attendee URL.

3.1.4.2.3 Complex Types

The following XML Schema complex type definitions are specific to this operation.

3.1.4.2.3.1 BroadcastUser

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

This complex type specifies a unique user of a slide show broadcast session.

```
<xs:complexType name="BroadcastUser">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="SessionId" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="UserToken" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

SessionId: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identification of the slide show broadcast session on the protocol server. This element **MUST** be present.

UserToken: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identification of a user of the slide show broadcast session on the protocol server. This element **MUST** be present.

3.1.4.2.4 Simple Types

None.

3.1.4.2.5 Attributes

None.

3.1.4.2.6 Groups

None.

3.1.4.2.7 Attribute Groups

None.

3.1.4.3 BroadcastGetHostToken

The **BroadcastGetHostToken** operation receives a presentation identifier as input and returns a string that represents the host token for that presentation.

```
<wsdl:operation name="BroadcastGetHostToken">
  <wsdl:input message="tns:BroadcastGetHostTokenSoapIn"/>
  <wsdl:output message="tns:BroadcastGetHostTokenSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastGetHostTokenSoapIn** request message and the protocol server **MUST** respond with a **BroadcastGetHostTokenSoapOut** response message as specified in the following sections.

3.1.4.3.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.3.1.1 BroadcastGetHostTokenSoapIn

The request WSDL message for the **BroadcastGetHostToken** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetHostToken
```

The SOAP body contains the **BroadcastGetHostToken** element.

3.1.4.3.1.2 BroadcastGetHostTokenSoapOut

The response WSDL message for the **BroadcastGetHostToken** WSDL operation.

The SOAP body contains the **BroadcastGetHostTokenResponse** element.

3.1.4.3.2 Elements

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

3.1.4.3.2.1 BroadcastGetHostToken

The input data for the **BroadcastGetHostToken** WSDL operation.

```
<xs:element name="BroadcastGetHostToken">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="presentationId" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

presentationId: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the identifier of the presentation for which to return a host token. This element **MUST** be present.

3.1.4.3.2.2 BroadcastGetHostTokenResponse

The result data for the **BroadcastGetHostToken** WSDL operation.

```
<xs:element name="BroadcastGetHostTokenResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastGetHostTokenResult"
type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastGetHostTokenResult: A [ServiceResult](#) that specifies the result of the operation. If the **Result** child element of **BroadcastGetHostTokenResult** is present, the **Result** child element **MUST** be an **s:string** ([\[XMLSCHEMA2\]](#) section 3.2.1).

3.1.4.3.3 Complex Types

None.

3.1.4.3.4 Simple Types

None.

3.1.4.3.5 Attributes

None.

3.1.4.3.6 Groups

None.

3.1.4.3.7 Attribute Groups

None.

3.1.4.4 BroadcastGetNewUploadFile

The **BroadcastGetNewUploadFile** operation returns a **BroadcastFile** (section [2.2.4.2](#)) that contains the upload URL and identifier for the presentation to be used during a slide show broadcast.

```
<wsdl:operation name="BroadcastGetNewUploadFile">
  <wsdl:input message="tns:BroadcastGetNewUploadFileSoapIn"/>
  <wsdl:output message="tns:BroadcastGetNewUploadFileSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastGetNewUploadFileSoapIn** request message and the protocol server MUST respond with a **BroadcastGetNewUploadFileSoapOut** response message as specified in the following sections.

3.1.4.4.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.4.1.1 BroadcastGetNewUploadFileSoapIn

The request WSDL message for the **BroadcastGetNewUploadFile** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetNewUploadFile
```

The SOAP body contains the **BroadcastGetNewUploadFile** element.

3.1.4.4.1.2 BroadcastGetNewUploadFileSoapOut

The response WSDL message for the **BroadcastGetNewUploadFile** WSDL operation.

The SOAP body contains the **BroadcastGetNewUploadFileResponse** element.

3.1.4.4.2 Elements

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

3.1.4.4.2.1 BroadcastGetNewUploadFile

The input data for the **BroadcastGetNewUploadFile** WSDL operation.

```
<xs:element name="BroadcastGetNewUploadFile">
  <xs:complexType/>
</xs:element>
```

3.1.4.4.2 BroadcastGetNewUploadFileResponse

The result data for the **BroadcastGetNewUploadFile** WSDL operation.

```
<xs:element name="BroadcastGetNewUploadFileResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastGetNewUploadFileResult"
        type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastGetNewUploadFileResult: A [ServiceResult](#) that specifies the result of the operation. If the **Result** child element of **BroadcastGetNewUploadFileResult** is present, then the **Result** child element MUST be a [BroadcastFile](#) element.

3.1.4.4.3 Complex Types

None.

3.1.4.4.4 Simple Types

None.

3.1.4.4.5 Attributes

None.

3.1.4.4.6 Groups

None.

3.1.4.4.7 Attribute Groups

None.

3.1.4.5 BroadcastGetServerInfo

The **BroadcastGetServerInfo** operation returns information about the protocol server.

```
<wsdl:operation name="BroadcastGetServerInfo">
  <wsdl:input message="tns:BroadcastGetServerInfoSoapIn"/>
  <wsdl:output message="tns:BroadcastGetServerInfoSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastGetServerInfoSoapIn** request message and the protocol server MUST respond with a **BroadcastGetServerInfoSoapOut** response message as specified in the following sections.

3.1.4.5.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.5.1.1 BroadcastGetServerInfoSoapIn

The request WSDL message for the **BroadcastGetServerInfo** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetServerInfo
```

The SOAP body contains the **BroadcastGetServerInfo** element.

3.1.4.5.1.2 BroadcastGetServerInfoSoapOut

The response WSDL message for the **BroadcastGetServerInfo** WSDL operation.

The SOAP body contains the **BroadcastGetServerInfoResponse** element.

3.1.4.5.2 Elements

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

3.1.4.5.2.1 BroadcastGetServerInfo

The input data for the **BroadcastGetServerInfo** WSDL operation.

```
<xs:element name="BroadcastGetServerInfo">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="clientInfo" type="tns:ClientInfo"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

clientInfo: A [ClientInfo](#) element that specifies information about the protocol client. This element MUST be present.

3.1.4.5.2.2 BroadcastGetServerInfoResponse

The result data for the **BroadcastGetServerInfo** WSDL operation.

```
<xs:element name="BroadcastGetServerInfoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastGetServerInfoResult"
type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastGetServerInfoResult: A [ServiceResult](#) that specifies the result of the operation. If the **Result** child element of **BroadcastGetServerInfoResult** is present, then the **Result** child element MUST be a [ServerInfo](#) element.

3.1.4.5.3 Complex Types

The following XML Schema complex type definitions are specific to this operation.

3.1.4.5.3.1 ClientInfo

Namespace: <http://schemas.microsoft.com/server/powerpoint/2009/main>

This complex type specifies the protocol client's name and version.

```
<xs:complexType name="ClientInfo">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="Name" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="Version" type="tns:Version"/>
  </xs:sequence>
</xs:complexType>
```

Name: An **s:string** element ([\[XMLSCHEMA2\]](#) section 3.2.1) that specifies the name of the protocol client. This element **MUST** be present.

Version: A [Version](#) element that specifies the version of the protocol client. This element **MUST** be present.

3.1.4.5.4 Simple Types

None.

3.1.4.5.5 Attributes

None.

3.1.4.5.6 Groups

None.

3.1.4.5.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

3.2 Client Details

None.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Message Processing Events and Sequencing Rules

None.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

4 Protocol Examples

The following examples demonstrate the interaction between the protocol client and the protocol server and the PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server. For the sake of succinctness, only the SOAP body is listed.

4.1 Retrieving Protocol Server Information

The protocol client first retrieves the protocol server information by sending a **BroadcastGetServerInfoSoapIn** message to the protocol server.

```
<BroadcastGetServerInfo xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <clientInfo xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
    <Name>Microsoft PowerPoint</Name>
    <Version>
      <MajorNumber>14</MajorNumber>
      <MinorNumber>0</MinorNumber>
    </Version>
  </clientInfo>
</BroadcastGetServerInfo>
```

The protocol server responds by sending a **BroadcastGetServerInfoSoapOut** message. In the following example, the server does not support audio, video, notes and handout masters. The protocol client needs to strip out the unsupported content from the presentation before uploading it to the protocol server. Also, the protocol server specifies that the presentation size can't be greater than 20MB and that it expects the protocol client to use WebDAV PUT request as specified in [\[RFC2518\]](#) to upload the presentation.

```
<BroadcastGetServerInfoResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <BroadcastGetServerInfoResult>
    <Result xsi:type="ServerInfo">
      <Name>Broadcast Host</Name>
      <Version>
        <MajorNumber>14</MajorNumber>
        <MinorNumber>0</MinorNumber>
      </Version>
    <PresenterServiceUrl>http://www.contoso.com/sites/broadcast/_vti_bin/present.asmx?wsdl</PresenterServiceUrl>
    <CapabilitiesList>
      <CapabilityData>
        <Capability>SupportVideo</Capability>
        <Value>>false</Value>
      </CapabilityData>
      <CapabilityData>
        <Capability>UseWebDav</Capability>
        <Value>>true</Value>
      </CapabilityData>
      <CapabilityData>
        <Capability>SupportNotes</Capability>
        <Value>>false</Value>
      </CapabilityData>
      <CapabilityData>
        <Capability>SupportHandoutMaster</Capability>
        <Value>>false</Value>
      </CapabilityData>
    </CapabilitiesList>
  </BroadcastGetServerInfoResult>
</BroadcastGetServerInfoResponse>
```

```

    <CapabilityData>
      <Capability>SupportAudio</Capability>
      <Value>>false</Value>
    </CapabilityData>
  </CapabilitiesList>
</BroadcastGetServerInfoResponse>

```

4.2 Obtaining the Upload URL

The protocol client then sends **BroadcastGetNewUploadFileSoapIn** message to the protocol server to get the upload URL.

```
<BroadcastGetNewUploadFile xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main"/>
```

The protocol server sends the upload URL to the protocol client in **BroadcastGetNewUploadFileSoapOut** message. In the following example, the **FilePath** element contains the upload URL.

```

<BroadcastGetNewUploadFileResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <BroadcastGetNewUploadFileResult>
    <Result xsi:type="BroadcastFile">
      <FilePath>http://www.contoso.com/sites/broadcast/9d51d0b1f4774b6893cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</FilePath>
      <PresentationId>/9d51d0b1f4774b6893cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</PresentationId>
    </Result>
  </BroadcastGetNewUploadFileResult>
</BroadcastGetNewUploadFileResponse>

```

4.3 Uploading the Presentation to the Upload URL

The protocol client then uploads the presentation to the upload URL using WebDAV PUT request as specified in [RFC2518](#). Once the presentation has been uploaded, the protocol client sends **BroadcastGetHostTokenSoapIn** message to the protocol server. In the message it passes the **presentationId** that it received in the **BroadcastGetNewUploadFileSoapOut** message.

```
<BroadcastGetHostToken xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
```



```

    <presentationId
xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">/9d51d0b1f4774b6893
cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</presentationId>
</BroadcastGetHostToken>

```

The protocol server returns a host token corresponding to the **presentationId** in **BroadcastGetHostTokenSoapOut** message. Although in this example the protocol server simply returns the **presentationId** back, it doesn't need to. The protocol server can return any string it wants as long as the protocol server is able to retrieve the original presentation given the host token.

```

<BroadcastGetHostTokenResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <BroadcastGetHostTokenResult>
    <Result xsi:type="xsd:string">/9d51d0b1f4774b6893cb728c0ba15a57/fc5a959c-70a1-4fa9-
98a8-373781bd3fa9.pptx</Result>
  </BroadcastGetHostTokenResult>
</BroadcastGetHostTokenResponse>

```

4.4 Starting the Broadcast Session

The protocol client then sends **BroadcastStartSessionSoapIn** message to the PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server and passes in the host token it received in **BroadcastGetHostTokenSoapOut** message.

```

<BroadcastStartSession xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <hostToken
xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">/9d51d0b1f4774b6893
cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</hostToken>
</BroadcastStartSession>

```

PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server responds with the **BroadcastStartSessionSoapOut** message and sends in a BroadcastUser [\[MS-PWBPS\]](#) section 2.2.4.1 element which contains the slide show broadcast session identifier.

```

<BroadcastStartSessionResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <BroadcastStartSessionResult>
    <Result xsi:type="BroadcastUser">
      <SessionId>d3679f05-7d93-4b4c-82d5-92aa5c40ac96</SessionId>
      <UserToken>29163435-d9d8-46eb-aec8-2382c5c1596c</UserToken>
    </Result>
  </BroadcastStartSessionResult>
</BroadcastStartSessionResponse>

```

4.5 Retrieving the Attendee URL for the Broadcast

The protocol client then sends a **BroadcastGetAttendeeUrlSoapIn** message to the protocol server to retrieve the attendee URL to be distributed to the slide show broadcast attendees. It passes in the BroadcastUser [\[MS-PWBPS\]](#) section 2.2.4.1 element that it retrieved from the **BroadcastStartSessionSoapOut** message.

```

<BroadcastGetAttendeeUrl xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">

```

```

<user xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <SessionId>d3679f05-7d93-4b4c-82d5-92aa5c40ac96</SessionId>
  <UserToken>29163435-d9d8-46eb-aec8-2382c5c1596c</UserToken>
</user>
  <presentationId
xmlns:SOAPSDK5="http://schemas.microsoft.com/server/powerpoint/2009/main">/9d51d0b1f4774b6893
cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</presentationId>
</BroadcastGetAttendeeUrl>

```

The protocol server returns the attendee URL in the **BroadcastGetAttendeeUrlSoapOut** message.

```

<BroadcastGetAttendeeUrlResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <BroadcastGetAttendeeUrlResult>
    <Result
xsi:type="xsd:string">http://www.contoso.com/sites/broadcast/_layouts/PowerPointFrame.aspx?Po
werPointView=AttendeeView&S3SessionId=d3679f05-7d93-4b4c-82d5-92aa5c40ac96</Result>
    </BroadcastGetAttendeeUrlResult>
  </BroadcastGetAttendeeUrlResult>
</BroadcastGetAttendeeUrlResponse>

```

4.6 Using the BroadcastPutDataSoapIn Message

During the duration of the slide show broadcast session, the protocol client sends **BroadcastPutDataSoapIn** messages to the PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server.

```

<BroadcastPutData xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <user xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
    <SessionId>cca96f89-0718-47d5-ad26-27c17d2cc9ce</SessionId>
    <UserToken>f12df8ef-8f76-42ab-ac49-59205b6d5f25</UserToken>
  </user>
  <data xmlns:SOAPSDK5="http://schemas.microsoft.com/server/powerpoint/2009/main">
    <SlideShowState>NotStartedYet</SlideShowState>
    <HostToken>/9d51d0b1f4774b6893cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-
373781bd3fa9.pptx</HostToken>
    <SlideId>256</SlideId>
    <LaserPointer><X>0</X><Y>0</Y></LaserPointer>
    <AnimationStepDataList/><LaserPointerDataList/>
    <SequenceNumber>1</SequenceNumber>
  </data>
</BroadcastPutData>

```

4.7 Ending the Broadcast Session

When it's time to end the slide show broadcast session, the protocol client sends a **BroadcastEndSessionSoapIn** message to the PowerPoint Web Broadcast Protocol [\[MS-PWBPS\]](#) server.

```

<BroadcastEndSession xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <user xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
    <SessionId>d3679f05-7d93-4b4c-82d5-92aa5c40ac96</SessionId>
    <UserToken>29163435-d9d8-46eb-aec8-2382c5c1596c</UserToken>
  </user>
</BroadcastEndSession>

```

4.8 Deleting the Uploaded Presentation

Finally, the protocol client sends a **BroadcastDeleteUploadFileSoapIn** message to the protocol server to delete the uploaded presentation.

```
<BroadcastDeleteUploadFile xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
  <presentationId
xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">/9d51d0b1f4774b6893
cb728c0ba15a57/fc5a959c-70a1-4fa9-98a8-373781bd3fa9.pptx</presentationId>
</BroadcastDeleteUploadFile>
```

5 Security

5.1 Security Considerations for Implementers

Protocol servers can authenticate all file upload requests. Additionally, the protocol servers can impose a maximum file size limit on the uploaded files. Protocol servers can communicate this limit to the protocol clients by setting the **FileSizeLimited** [ServerCapability](#) in the implementation of [BroadcastGetServerInfo](#) web operation. Protocol servers can also cleanup old files that are not deleted by the protocol clients.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL is provided:

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://schemas.microsoft.com/server/powerpoint/2009/main"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://schemas.microsoft.com/server/powerpoint/2009/main"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/server/powerpoint/2009/main">
      <s:element name="BroadcastGetNewUploadFile">
        <s:complexType />
      </s:element>
      <s:element name="BroadcastGetNewUploadFileResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="BroadcastGetNewUploadFileResult"
type="tns:ServiceResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ServiceResult">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Result" />
          <s:element minOccurs="0" maxOccurs="1" name="Error" type="tns:ServiceError" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ServiceError">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Message" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Title" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Type" type="tns:ServiceErrorType" />
          <s:element minOccurs="1" maxOccurs="1" name="RecommendedActions"
type="tns:ClientActions" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="ServiceErrorType">
        <s:restriction base="s:string">
          <s:enumeration value="UnknownError" />
          <s:enumeration value="ApplicationError" />
          <s:enumeration value="Timeout" />
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="ClientActions">
        <s:list>
          <s:simpleType>
            <s:restriction base="s:string">
              <s:enumeration value="None" />
              <s:enumeration value="Dismiss" />
              <s:enumeration value="Close" />
              <s:enumeration value="OpenInClient" />
            </s:restriction>
          </s:simpleType>
        </s:list>
      </s:simpleType>
    </s:schema>
  </wsdl:types>

```

```

        <s:enumeration value="Refresh" />
    </s:restriction>
</s:simpleType>
</s:list>
</s:simpleType>
<s:complexType name="ServerInfo">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="Version" type="tns:Version" />
        <s:element minOccurs="0" maxOccurs="1" name="PresenterServiceUrl" type="s:string"
/>
    <s:element minOccurs="0" maxOccurs="1" name="CapabilitiesList"
type="tns:ArrayOfCapabilityData" />
    </s:sequence>
</s:complexType>
<s:complexType name="Version">
    <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="MajorNumber" type="s:int" />
        <s:element minOccurs="1" maxOccurs="1" name="MinorNumber" type="s:int" />
    </s:sequence>
</s:complexType>
<s:complexType name="ArrayOfCapabilityData">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="CapabilityData"
nillable="true" type="tns:CapabilityData" />
    </s:sequence>
</s:complexType>
<s:complexType name="CapabilityData">
    <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="Capability"
type="tns:ServerCapability" />
        <s:element minOccurs="0" maxOccurs="1" name="Value" type="s:string" />
    </s:sequence>
</s:complexType>
<s:simpleType name="ServerCapability">
    <s:restriction base="s:string">
        <s:enumeration value="SupportAudio" />
        <s:enumeration value="SupportVideo" />
        <s:enumeration value="SupportNotes" />
        <s:enumeration value="SupportHandoutMaster" />
        <s:enumeration value="FileSizeLimited" />
        <s:enumeration value="UseWebDav" />
        <s:enumeration value="SessionTimeout" />
        <s:enumeration value="SessionIdleTimeout" />
    </s:restriction>
</s:simpleType>
<s:complexType name="BroadcastFile">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="FilePath" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="PresentationId" type="s:string" />
    </s:sequence>
</s:complexType>
<s:element name="BroadcastGetHostToken">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="presentationId" type="s:string" />
        </s:sequence>
    </s:complexType>
</s:element>

```

```

<s:element name="BroadcastGetHostTokenResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="BroadcastGetHostTokenResult"
type="tns:ServiceResult" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="BroadcastGetAttendeeUrl">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="user" type="tns:BroadcastUser" />
      <s:element minOccurs="0" maxOccurs="1" name="presentationId" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="BroadcastUser">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="SessionId" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="UserToken" type="s:string" />
  </s:sequence>
</s:complexType>
<s:element name="BroadcastGetAttendeeUrlResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="BroadcastGetAttendeeUrlResult"
type="tns:ServiceResult" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="BroadcastDeleteUploadFile">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="presentationId" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="BroadcastDeleteUploadFileResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="BroadcastDeleteUploadFileResult"
type="tns:ServiceResult" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="BroadcastGetServerInfo">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="clientInfo" type="tns:ClientInfo" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ClientInfo">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="Version" type="tns:Version" />
  </s:sequence>
</s:complexType>
<s:element name="BroadcastGetServerInfoResponse">

```

```

        <s:complexType>
            <s:sequence>
                <s:element minOccurs="0" maxOccurs="1" name="BroadcastGetServerInfoResult"
type="tns:ServiceResult" />
            </s:sequence>
        </s:complexType>
    </s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="BroadcastGetNewUploadFileSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastGetNewUploadFile" />
</wsdl:message>
<wsdl:message name="BroadcastGetNewUploadFileSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastGetNewUploadFileResponse" />
</wsdl:message>
<wsdl:message name="BroadcastGetHostTokenSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastGetHostToken" />
</wsdl:message>
<wsdl:message name="BroadcastGetHostTokenSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastGetHostTokenResponse" />
</wsdl:message>
<wsdl:message name="BroadcastGetAttendeeUrlSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastGetAttendeeUrl" />
</wsdl:message>
<wsdl:message name="BroadcastGetAttendeeUrlSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastGetAttendeeUrlResponse" />
</wsdl:message>
<wsdl:message name="BroadcastDeleteUploadFileSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastDeleteUploadFile" />
</wsdl:message>
<wsdl:message name="BroadcastDeleteUploadFileSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastDeleteUploadFileResponse" />
</wsdl:message>
<wsdl:message name="BroadcastGetServerInfoSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastGetServerInfo" />
</wsdl:message>
<wsdl:message name="BroadcastGetServerInfoSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastGetServerInfoResponse" />
</wsdl:message>
<wsdl:portType name="PowerpointBroadcastHostWebService_1_0Soap">
    <wsdl:operation name="BroadcastGetNewUploadFile">
        <wsdl:input message="tns:BroadcastGetNewUploadFileSoapIn" />
        <wsdl:output message="tns:BroadcastGetNewUploadFileSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetHostToken">
        <wsdl:input message="tns:BroadcastGetHostTokenSoapIn" />
        <wsdl:output message="tns:BroadcastGetHostTokenSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetAttendeeUrl">
        <wsdl:input message="tns:BroadcastGetAttendeeUrlSoapIn" />
        <wsdl:output message="tns:BroadcastGetAttendeeUrlSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastDeleteUploadFile">
        <wsdl:input message="tns:BroadcastDeleteUploadFileSoapIn" />
        <wsdl:output message="tns:BroadcastDeleteUploadFileSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetServerInfo">
        <wsdl:input message="tns:BroadcastGetServerInfoSoapIn" />
        <wsdl:output message="tns:BroadcastGetServerInfoSoapOut" />
    </wsdl:operation>

```



```

    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="PowerpointBroadcastHostWebService_1_0Soap"
type="tns:PowerpointBroadcastHostWebService_1_0Soap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="BroadcastGetNewUploadFile">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetNewUploadFil
e" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetHostToken">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetHostToken"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetAttendeeUrl">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetAttendeeUrl"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastDeleteUploadFile">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastDeleteUploadFil
e" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastGetServerInfo">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetServerInfo"
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="PowerpointBroadcastHostWebService_1_0Soap12"
type="tns:PowerpointBroadcastHostWebService_1_0Soap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="BroadcastGetNewUploadFile">
    <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetNewUploadFil
e" style="document" />
    <wsdl:input>
      <soap12:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="BroadcastGetHostToken">
    <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetHostToken"
style="document" />
    <wsdl:input>
      <soap12:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="BroadcastGetAttendeeUrl">
    <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetAttendeeUrl"
style="document" />
    <wsdl:input>
      <soap12:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="BroadcastDeleteUploadFile">
    <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastDeleteUploadFil
e" style="document" />
    <wsdl:input>
      <soap12:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap12:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="BroadcastGetServerInfo">
    <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastGetServerInfo"
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® PowerPoint® 2010
- Microsoft® PowerPoint® Web App

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

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

9 Index

A

Abstract data model
 [client](#) 29
 [server](#) 19
[Applicability](#) 8
[ArrayOfCapabilityData complex type](#) 10
[Attribute groups](#) 16
[Attributes](#) 16

B

[BroadcastFile complex type](#) 10

C

[Capability negotiation](#) 8
[CapabilityData complex type](#) 11
[Change tracking](#) 45

Client

[abstract data model](#) 29
 [details](#) 29
 [initialization](#) 30
 [local events](#) 30
 [message processing](#) 30
 [overview](#) 17
 [sequencing rules](#) 30
 [timer events](#) 30
 [timers](#) 30
[ClientActions simple type](#) 14
[Complex types](#) 10
 [ArrayOfCapabilityData](#) 10
 [BroadcastFile](#) 10
 [CapabilityData](#) 11
 [ServerInfo](#) 12
 [ServiceError](#) 12
 [ServiceResult](#) 13
 [Version](#) 13

D

Data model - abstract
 [client](#) 29
 [server](#) 19
[Deleting the uploaded presentation example](#) 35

E

[Ending the broadcast session example](#) 34

Events

[local - client](#) 30
 [local - server](#) 29
 [timer - client](#) 30
 [timer - server](#) 29

Example

[obtaining the upload URL](#) 32

Examples

[deleting the uploaded presentation](#) 35
 [ending the broadcast session](#) 34

[overview](#) 31
 [retrieving protocol server information](#) 31
 [retrieving the attendee URL for the broadcast](#) 33
 [starting the broadcast session](#) 33
 [uploading the presentation to the upload URL](#) 32
 [using the BroadcastPutDataSoapIn message](#) 34

F

[Fields - vendor-extensible](#) 8
[Full WSDL](#) 37

G

[Glossary](#) 6
[Groups](#) 16

I

[Implementer - security considerations](#) 36
[Index of security parameters](#) 36
[Informative references](#) 7

Initialization

[client](#) 30
 [server](#) 20
[Introduction](#) 6

L

Local events
 [client](#) 30
 [server](#) 29

M

Message processing
 [client](#) 30
 [server](#) 20

Messages

[ArrayOfCapabilityData complex type](#) 10
 [attribute groups](#) 16
 [attributes](#) 16
 [BroadcastFile complex type](#) 10
 [CapabilityData complex type](#) 11
 [ClientActions simple type](#) 14
 [complex types](#) 10
 [elements](#) 9
 [enumerated](#) 9
 [groups](#) 16
 [namespaces](#) 9
 [ServerCapability simple type](#) 15
 [ServerInfo complex type](#) 12
 [ServiceError complex type](#) 12
 [ServiceErrorType simple type](#) 15
 [ServiceResult complex type](#) 13
 [simple types](#) 14
 [syntax](#) 9
 [transport](#) 9
 [Version complex type](#) 13

N

[Namespaces](#) 9
[Normative references](#) 6

O

[Obtaining the upload URL example](#) 32

Operations

[BroadcastDeleteUploadFile](#) 20
[BroadcastGetAttendeeUrl](#) 22
[BroadcastGetHostToken](#) 24
[BroadcastGetNewUploadFile](#) 26
[BroadcastGetServerInfo](#) 27

[Overview \(synopsis\)](#) 7

P

[Parameters - security index](#) 36

[Preconditions](#) 8

[Prerequisites](#) 8

[Product behavior](#) 44

R

[References](#) 6

[informative](#) 7

[normative](#) 6

[Relationship to other protocols](#) 8

[Retrieving protocol server information example](#) 31

[Retrieving the attendee URL for the broadcast example](#) 33

S

Security

[implementer considerations](#) 36

[parameter index](#) 36

Sequencing rules

[client](#) 30

[server](#) 20

Server

[abstract data model](#) 19

[BroadcastDeleteUploadFile operation](#) 20

[BroadcastGetAttendeeUrl operation](#) 22

[BroadcastGetHostToken operation](#) 24

[BroadcastGetNewUploadFile operation](#) 26

[BroadcastGetServerInfo operation](#) 27

[details](#) 17

[initialization](#) 20

[local events](#) 29

[message processing](#) 20

[overview](#) 17

[sequencing rules](#) 20

[timer events](#) 29

[timers](#) 19

[ServerCapability simple type](#) 15

[ServerInfo complex type](#) 12

[ServiceError complex type](#) 12

[ServiceErrorType simple type](#) 15

[ServiceResult complex type](#) 13

[Simple types](#) 14

[ClientActions](#) 14

[ServerCapability](#) 15

[ServiceErrorType](#) 15

[Standards assignments](#) 8

[Starting the broadcast session example](#) 33

Syntax

[messages - overview](#) 9

T

Timer events

[client](#) 30

[server](#) 29

Timers

[client](#) 30

[server](#) 19

[Tracking changes](#) 45

[Transport](#) 9

Types

[complex](#) 10

[simple](#) 14

U

[Uploading the presentation to the upload URL](#)

[example](#) 32

[Using the BroadcastPutDataSoapIn message](#)

[example](#) 34

V

[Vendor-extensible fields](#) 8

[Version complex type](#) 13

[Versioning](#) 8

W

[WSDL](#) 37