[MS-PWBPS]:

PowerPoint Web Broadcast Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
7/13/2009	0.1	Major	Initial Availability
8/28/2009	0.2	Editorial	Revised and edited the technical content
11/6/2009	0.3	Editorial	Revised and edited the technical content
2/19/2010	1.0	Major	Updated and revised the technical content
3/31/2010	1.01	Editorial	Revised and edited the technical content
4/30/2010	1.02	Editorial	Revised and edited the technical content
6/7/2010	1.03	Editorial	Revised and edited the technical content
6/29/2010	1.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	1.05	Minor	Clarified the meaning of the technical content.
9/27/2010	1.06	Minor	Clarified the meaning of the technical content.
11/15/2010	1.06	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.07	Editorial	Changed language and formatting in the technical content.
3/18/2011	1.07	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	1.07	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	1.07	None	No changes to the meaning, language, or formatting of the technical content.
4/11/2012	1.07	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	1.07	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	1.07	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2013	1.07	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	1.07	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	1.07	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	1.07	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	1.8	Minor	Clarified the meaning of the technical content.
7/31/2014	1.8	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
10/30/2014	1.8	None	No changes to the meaning, language, or formatting of the technical content.
6/23/2016	1.8	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document specifies the PowerPoint Web Broadcast Service Protocol, which enables a protocol client to update information about a **slide show broadcast** 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:

presentation slide: A slide that contains the content that can be displayed during a slide show. A presentation slide can derive formatting and content from a main master slide or a title master slide.

slide show: A delivery of a sequence of presentation slides, typically to an audience.

slide show broadcast: A delivery of a sequence of presentation slides, typically to an audience, as a single session between a protocol server and one or more protocol clients.

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.

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

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

website: A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as site.

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

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

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

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

1.2 References

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

1.2.1 Normative References

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

[MS-PWBHPS] Microsoft Corporation, "PowerPoint Web Broadcast Host Protocol".

[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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.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., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.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

None.

1.3 Protocol Overview (Synopsis)

This protocol enables a protocol client to send requests to a protocol server allowing the client to begin or end a slide show broadcast session, and to store data about the state of a broadcast on the protocol server.

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:

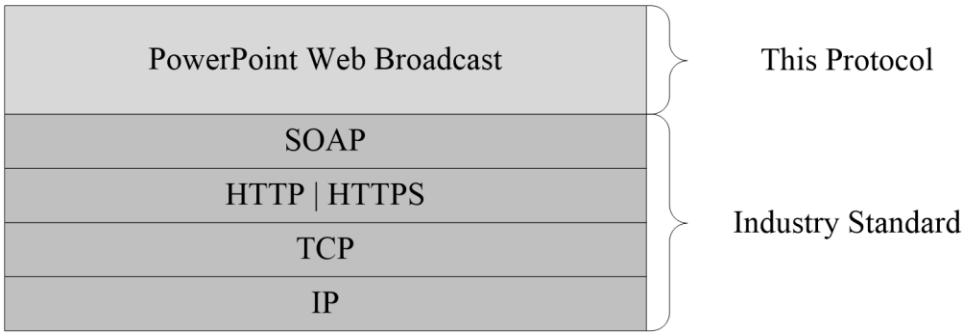

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

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

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

1.6 Applicability Statement

This protocol is designed to store slide show broadcast information 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 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

None.

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
BroadcastUser	A complex type that specifies an identifier for a user of the broadcast session.
ServiceError	The **ServiceError** type specifies error information returned by the protocol server to a protocol client.
ServiceResult	The **ServiceResult** type 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.

2.2.4.1 BroadcastUser

A complex type that specifies a user of a 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 xs:string [XMLSCHEMA2] section 3.2.1 element that specifies the identifier of the broadcast session on the protocol server. This element MUST be present.

UserToken: An xs:string [XMLSCHEMA2] section 3.2.1 element that specifies the identifier of a user of the broadcast session on the protocol server. This element MUST be present.

2.2.4.2 ServiceError

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 xs:string [XMLSCHEMA2] section 3.2.1 element 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 xs:string [XMLSCHEMA2] section 3.2.1 element 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.

2.2.4.3 ServiceResult

A complex type that specifies the result of a protocol method. 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 xs:anyType [XMLSCHEMA1] section 3.4.7 element 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.

2.2.5 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.
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

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 ServiceErrorType

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

None.

2.2.7 Groups

None.

2.2.8 Attribute Groups

None.

2.3 Directory Service Schema Elements

None.

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.

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC2616] (Section 10, Status Code Definitions).

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 high-level sequence diagrams illustrate the operation of the presenter client protocol.

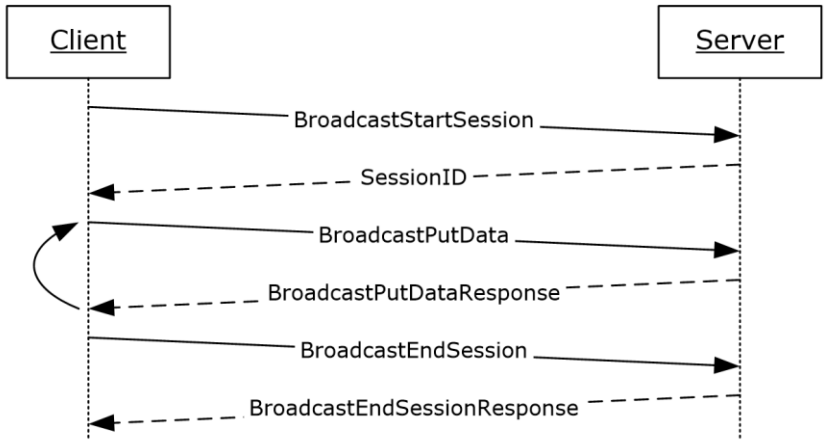

Figure 1: PowerPoint Web Broadcast Protocol high-level sequence diagram for presenter clients.

First, a protocol client acting as slide show broadcast presenter sends a BroadcastStartSession message and the protocol server responds with a BroadcastStartSessionResponse message containing the broadcast session identifier to be used for future requests. Next, the protocol client sends one or more BroadcastPutData messages containing the current state of the broadcast, and the server sends a BroadcastPutDataResponse message to acknowledge the

request. When the slide show broadcast is finished, the protocol client then sends a BroadcastEndSession message and the protocol server responds with a BroadcastEndSessionResponse message to acknowledge the request.

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.

SessionId: An entity that represents a unique identifier for a broadcast session.

hostToken: An entity that specifies the token returned by PowerPoint Web Broadcast Host protocol [MS-PWBHPS] server corresponding to the presentation that is uploaded by the protocol client.

Slide Show State: An entity that represents the current **slide show** state of the protocol client. It contains information such as the current slide and the current animation step of the protocol client.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

Section 3.1 specifies the sequencing of the protocol messages and how they relate to each other. The following sections specify the details of each individual message.

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

WSDL Operation	Description
BroadcastEndSession	The **BroadcastEndSession** operation is used by the presenter to end a broadcast session on the protocol server.
BroadcastPutData	The **BroadcastPutData** operation is used by the broadcast presenter to modify the current state of the broadcast session on the protocol server.
BroadcastStartSession	The **BroadcastStartSession** operation is used by the presenter to begin a broadcast session on the protocol server.

3.1.4.1 BroadcastEndSession

The **BroadcastEndSession** operation is used by the presenter to end a broadcast session on the protocol server.

```
<wsdl:operation name="BroadcastEndSession">
  <wsdl:input message="tns:BroadcastEndSessionSoapIn"/>
  <wsdl:output message="tns:BroadcastEndSessionSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastEndSessionSoapIn** request message, as specified in section 3.1.4.1.1.1, and the protocol server MUST respond with a **BroadcastEndSessionSoapOut** response message, as specified in section 3.1.4.1.1.2.

3.1.4.1.1 Messages

3.1.4.1.1.1 BroadcastEndSessionSoapIn

The requested WSDL message for a **BroadcastEndSession** WSDL operation.

The **SOAP action** value is:

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

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

3.1.4.1.1.2 BroadcastEndSessionSoapOut

The response WSDL message for a **BroadcastEndSession** method.

The SOAP action value is:

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

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

3.1.4.1.2 Elements

3.1.4.1.2.1 BroadcastEndSession

The input data for a **BroadcastEndSession** WSDL operation.

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

user: A BroadcastUser element that is obtained by making a BroadcastStartSession web method call. This element MUST be present.

3.1.4.1.2.2 BroadcastEndSessionResponse

The result data for a **BroadcastEndSession** WSDL operation.

```
<xs:element name="BroadcastEndSessionResponse">
  <xs:complexType>
    <xs:sequence>
```

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

BroadcastEndSessionResult: 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. The protocol client MAY retry the request or display the error to the user if the **Error** child element is present.

3.1.4.2 BroadcastPutData

The **BroadcastPutData** operation is used by the broadcast presenter to modify the current state of the broadcast session on the protocol server.

```
<wsdl:operation name="BroadcastPutData">
  <wsdl:input message="tns:BroadcastPutDataSoapIn"/>
  <wsdl:output message="tns:BroadcastPutDataSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastPutDataSoapIn** request message, and the protocol server MUST respond with a **BroadcastPutDataSoapOut** response message as follows:

3.1.4.2.1 Messages

3.1.4.2.1.1 BroadcastPutDataSoapIn

The requested WSDL message for a **BroadcastPutData** WSDL operation.

The SOAP action value is:

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

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

3.1.4.2.1.2 BroadcastPutDataSoapOut

The response WSDL message for a **BroadcastPutData** method.

The SOAP action value is:

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

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

3.1.4.2.2 Elements

3.1.4.2.2.1 BroadcastPutData

The input data for a **BroadcastPutData** WSDL operation.

```
<xs:element name="BroadcastPutData">
  <xs:complexType>
```

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

user: A BroadcastUser element that is obtained by making a BroadcastStartSession web method call. This element MUST be present.

data: A BroadcastData element that specifies the current slide show state on the protocol client. This element MUST be present. Protocol server MUST update its local copy of the state to match the protocol client state.

3.1.4.2.2.2 BroadcastPutDataResponse

The result data for a **BroadcastPutData** WSDL operation.

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

BroadcastPutDataResult: 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. The protocol client MAY retry the request or display the error to the user if the **Error** child element is present.

3.1.4.2.3 Complex Types

3.1.4.2.3.1 BroadcastData

A complex type that specifies data about the state of the broadcast.

```
<xs:complexType name="BroadcastData">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="SlideShowState" type="tns:SlideShowState"/>
    <xs:element minOccurs="0" maxOccurs="1" name="HostToken" type="xs:string"/>
    <xs:element minOccurs="1" maxOccurs="1" name="SlideId" type="xs:unsignedInt"/>
    <xs:element minOccurs="0" maxOccurs="1" name="AnimationStepDataList"
type="tns:ArrayOfBroadcastAnimationStepData"/>
    <xs:element minOccurs="1" maxOccurs="1" name="SequenceNumber" type="xs:int"/>
  </xs:sequence>
</xs:complexType>
```

SlideShowState: A SlideShowState element that specifies the current state of the slide show. This element MUST be present.

HostToken: An xs:string [XMLSCHEMA2] section 3.2.1 element that specifies the token returned by PowerPoint Web Broadcast Host protocol [MS-PWBHPS] server corresponding to the presentation that is uploaded by the protocol client. This element MUST be present.

SlideId: An xs:unsignedInt [XMLSCHEMA2] section 3.3.22 element that specifies the identifier of the **presentation slide**. This element MUST be present.

AnimationStepDataList: An ArrayOfBroadcastAnimationStepData element that specifies the current step in each of the animation timelines. This element MUST be present.

SequenceNumber: An xs:int [XMLSCHEMA2] section 3.3.17 element that specifies a monotonically increasing sequence number. This element MUST be present.

3.1.4.2.3.2 ArrayOfBroadcastAnimationStepData

A complex type that specifies a list of BroadcastAnimationStepData elements.

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

BroadcastAnimationStepData: Each element MUST specify a BroadcastAnimationStepData.

3.1.4.2.3.3 BroadcastAnimationStepData

A complex type that specifies a step in an animation timeline.

```
<xs:complexType name="BroadcastAnimationStepData">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="TimelineId" type="xs:string"/>
    <xs:element minOccurs="1" maxOccurs="1" name="Step" type="xs:int"/>
  </xs:sequence>
</xs:complexType>
```

TimelineId: An xs:string [XMLSCHEMA2] section 3.2.1 element that specifies an identifier of the animation timeline. MUST be present.

Step: An xs:int [XMLSCHEMA2] section 3.3.17 element that specifies the step number in the given animation timeline. MUST be present.

3.1.4.2.4 Simple Types

3.1.4.2.4.1 SlideShowState

A simple type that specifies an enumeration of all the possible slide show states.

```
<xs:simpleType name="SlideShowState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="NotStartedYet"/>
    <xs:enumeration value="BlackScreen"/>
    <xs:enumeration value="WhiteScreen"/>
    <xs:enumeration value="Normal"/>
    <xs:enumeration value="BroadcastEnded"/>
    <xs:enumeration value="SlideShowEnded"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for SlideShowState:

Value	Meaning
NotStartedYet	slide show is not started yet.
BlackScreen	slide show is displaying a black Screen.
WhiteScreen	slide show is displaying a white Screen.
Normal	slide show is displaying presentation slides.
BroadcastEnded	slide show broadcast has ended.
SlideShowEnded	slide show has ended.

3.1.4.3 BroadcastStartSession

The **BroadcastStartSession** operation is used by the presenter to begin a broadcast session on the protocol server.

```
<wsdl:operation name="BroadcastStartSession">
  <wsdl:input message="tns:BroadcastStartSessionSoapIn"/>
  <wsdl:output message="tns:BroadcastStartSessionSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastStartSessionSoapIn** request message, and the protocol server MUST respond with a **BroadcastStartSessionSoapOut** response message as follows:

3.1.4.3.1 Messages

3.1.4.3.1.1 BroadcastStartSessionSoapIn

The requested WSDL message for a **BroadcastStartSession** WSDL operation.

The SOAP action value is:

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

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

3.1.4.3.1.2 BroadcastStartSessionSoapOut

The response WSDL message for a **BroadcastStartSession** method.

The SOAP action value is:

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

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

3.1.4.3.2 Elements

3.1.4.3.2.1 BroadcastStartSession

The input data for a **BroadcastStartSession** WSDL operation.

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

hostToken: An xs:string [XMLSCHEMA2] section 3.2.1 element that specifies the token returned by PowerPoint Web Broadcast Host protocol [MS-PWBHPS] server corresponding to the presentation that is uploaded by the protocol client. This element MUST be present.

3.1.4.3.2.2 BroadcastStartSessionResponse

The result data for a **BroadcastStartSession** WSDL operation.

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

BroadcastStartSessionResult: A ServiceResult that specifies the result of the operation. This element MUST be present. The **Result** child element MUST be a BroadcastUser if the **Error** child element is not present.

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 contain sample interactions between protocol clients and protocol servers.

4.1 Presenter Client Example

The presenter protocol client begins by sending a request to the protocol server to begin the slide show broadcast. The following **BroadcastStartSessionSoapIn** message is sent to the protocol server:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <BroadcastStartSession
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <hostToken
xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
            /sites/broadcast/9d51d0b1f4774b6893cb728c0ba15a57/faf48352-f17f-46ed-ad44-
9adac3fd37bb.pptx
         </hostToken>
      </BroadcastStartSession>
   </soap:Body>
</soap:Envelope>
```

The protocol server responds with a message to acknowledge the request and to provide a user identifier and a session identifier to be used by the presenter protocol client for future requests. The following **BroadcastStartSessionSoapOut** message is sent to the presenter protocol client:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <BroadcastStartSessionResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <BroadcastStartSessionResult>
            <Result xsi:type="BroadcastUser">
               <SessionId>601022d2-306a-43fb-8ffd-a4a739cba8c0</SessionId>
               <UserToken>186bfc54-0f78-47e9-b994-d2f40aa91d66</UserToken>
            </Result>
         </BroadcastStartSessionResult>
      </BroadcastStartSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Next, the presenter protocol client sends information about the current state of the slide show broadcast. The following **BroadcastPutDataSoapIn** message is sent to the protocol server:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <BroadcastPutData xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <user xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
            <SessionId>601022d2-306a-43fb-8ffd-a4a739cba8c0</SessionId>
            <UserToken>186bfc54-0f78-47e9-b994-d2f40aa91d66</UserToken>
         </user>
         <data>
            <SlideShowState>Normal</SlideShowState>
            <HostToken>/sites/broadcast/9d51d0b1f4774b6893cb728c0ba15a57/faf48352-f17f-46ed-
ad44-9adac3fd37bb.pptx</HostToken>
            <SlideId>256</SlideId>
            <AnimationStepDataList>
               <BroadcastAnimationStepData>
```

```
                  <TimelineId>0_anim</TimelineId>
                  <Step>0</Step>
               </BroadcastAnimationStepData>
            </AnimationStepDataList>
            <SequenceNumber>2</SequenceNumber>
         </data>
      </BroadcastPutData>
   </soap:Body>
</soap:Envelope>
```

The protocol server responds with a message to acknowledge the request. The following **BroadcastPutDataSoapOut** message is sent to the presenter protocol client:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <BroadcastPutDataResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <BroadcastPutDataResult />
      </BroadcastPutDataResponse>
   </soap:Body>
</soap:Envelope>
```

The presenter protocol client continues to send these **BroadcastPutDataSoapIn** messages until the slide show broadcast is finished. At this point, the presenter protocol client sends a request to the protocol server to end the slide show broadcast. The following **BroadcastEndSessionSoapIn** message is sent to the protocol server:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Body>
      <BroadcastEndSession xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <user xmlns:SOAPSDK4="http://schemas.microsoft.com/server/powerpoint/2009/main">
            <SessionId>601022d2-306a-43fb-8ffd-a4a739cba8c0</SessionId>
            <UserToken>186bfc54-0f78-47e9-b994-d2f40aa91d66</UserToken>
         </user>
      </BroadcastEndSession>
   </soap:Body>
</soap:Envelope>
```

The protocol server responds with a message to acknowledge the request. The following **BroadcastEndSessionSoapOut** message is sent to the presenter protocol client:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <BroadcastEndSessionResponse
xmlns="http://schemas.microsoft.com/server/powerpoint/2009/main">
         <BroadcastEndSessionResult />
      </BroadcastEndSessionResponse>
   </soap:Body>
</soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

There are no security considerations that are specific to this protocol. General security considerations pertaining to [RFC2822] apply.

This protocol does not introduce any additional security considerations beyond those that apply to its underlying protocols.

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="BroadcastStartSession">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="hostToken" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="BroadcastStartSessionResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="BroadcastStartSessionResult"
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:enumeration value="Refresh" />
            </s:restriction>
          </s:simpleType>
        </s:list>
```

```
      </s:simpleType>
      <s:element name="BroadcastEndSession">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="user" type="tns:BroadcastUser" />
          </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="BroadcastEndSessionResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="BroadcastEndSessionResult"
type="tns:ServiceResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="BroadcastPutData">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="user" type="tns:BroadcastUser" />
            <s:element minOccurs="0" maxOccurs="1" name="data" type="tns:BroadcastData" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="BroadcastData">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SlideShowState"
type="tns:SlideShowState" />
          <s:element minOccurs="0" maxOccurs="1" name="HostToken" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="SlideId" type="s:unsignedInt" />
          <s:element minOccurs="0" maxOccurs="1" name="AnimationStepDataList"
type="tns:ArrayOfBroadcastAnimationStepData" />
          <s:element minOccurs="1" maxOccurs="1" name="SequenceNumber" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="SlideShowState">
        <s:restriction base="s:string">
          <s:enumeration value="NotStartedYet" />
          <s:enumeration value="BlackScreen" />
          <s:enumeration value="WhiteScreen" />
          <s:enumeration value="Normal" />
          <s:enumeration value="BroadcastEnded" />
          <s:enumeration value="SlideShowEnded" />
        </s:restriction>
      </s:simpleType>
      <s:complexType name="ArrayOfBroadcastAnimationStepData">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="BroadcastAnimationStepData"
nillable="true" type="tns:BroadcastAnimationStepData" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="BroadcastAnimationStepData">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="TimelineId" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Step" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:element name="BroadcastPutDataResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="BroadcastPutDataResult"
type="tns:ServiceResult" />
          </s:sequence>
```

```
        </s:complexType>
      </s:element>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="BroadcastStartSessionSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastStartSession" />
  </wsdl:message>
  <wsdl:message name="BroadcastStartSessionSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastStartSessionResponse" />
  </wsdl:message>
  <wsdl:message name="BroadcastEndSessionSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastEndSession" />
  </wsdl:message>
  <wsdl:message name="BroadcastEndSessionSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastEndSessionResponse" />
  </wsdl:message>
  <wsdl:message name="BroadcastPutDataSoapIn">
    <wsdl:part name="parameters" element="tns:BroadcastPutData" />
  </wsdl:message>
  <wsdl:message name="BroadcastPutDataSoapOut">
    <wsdl:part name="parameters" element="tns:BroadcastPutDataResponse" />
  </wsdl:message>
  <wsdl:portType name="PptPresentServiceSoap">
    <wsdl:operation name="BroadcastStartSession">
      <wsdl:input message="tns:BroadcastStartSessionSoapIn" />
      <wsdl:output message="tns:BroadcastStartSessionSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastEndSession">
      <wsdl:input message="tns:BroadcastEndSessionSoapIn" />
      <wsdl:output message="tns:BroadcastEndSessionSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BroadcastPutData">
      <wsdl:input message="tns:BroadcastPutDataSoapIn" />
      <wsdl:output message="tns:BroadcastPutDataSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="PptPresentServiceSoap" type="tns:PptPresentServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="BroadcastStartSession">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastStartSession"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastEndSession">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastEndSession"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastPutData">
      <soap:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastPutData"
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="PptPresentServiceSoap12" type="tns:PptPresentServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="BroadcastStartSession">
      <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastStartSession"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastEndSession">
      <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastEndSession"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BroadcastPutData">
      <soap12:operation
soapAction="http://schemas.microsoft.com/server/powerpoint/2009/main/BroadcastPutData"
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

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

S

T

V

W