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.

- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.

- **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.

**Support.** For questions and support, please contact dochelp@microsoft.com.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/20/2012</td>
<td>0.1</td>
<td>New</td>
<td>Released new document.</td>
</tr>
<tr>
<td>4/11/2012</td>
<td>0.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>0.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>1.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/30/2013</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>1.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>1.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>6/23/2016</td>
<td>1.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>1.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/19/2017</td>
<td>1.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>10/1/2018</td>
<td>2.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>5/19/2020</td>
<td>2.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>8/18/2020</td>
<td>2.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction ........................................................................................................... 6
  1.1 Glossary ............................................................................................................. 6
  1.2 References ......................................................................................................... 7
    1.2.1 Normative References ............................................................................... 7
    1.2.2 Informative References ............................................................................. 8
  1.3 Overview ........................................................................................................... 8
  1.4 Relationship to Other Protocols ....................................................................... 8
  1.5 Prerequisites/Preconditions .............................................................................. 9
  1.6 Applicability Statement .................................................................................... 9
  1.7 Versioning and Capability Negotiation ............................................................. 9
  1.8 Vendor-Extensible Fields .................................................................................. 9
  1.9 Standards Assignments ..................................................................................... 9

2 Messages .............................................................................................................. 10
  2.1 Transport .......................................................................................................... 10
  2.2 Common Message Syntax ............................................................................... 10
    2.2.1 Namespaces ............................................................................................... 10
    2.2.2 Messages .................................................................................................... 10
    2.2.3 Elements .................................................................................................... 10
    2.2.4 Complex Types ......................................................................................... 11
      2.2.4.1 AppServerInfo .................................................................................. 11
      2.2.4.2 ArrayOfPPTBroadcastAnimationStepData ...................................... 12
      2.2.4.3 ArrayOfPPTBroadcastMediaStateData ............................................. 12
      2.2.4.4 BroadcastUser ................................................................................... 12
      2.2.4.5 EnumTypes ....................................................................................... 13
      2.2.4.6 PPTBroadcastAnimationStepData .................................................. 13
      2.2.4.7 PPTBroadcastMediaStateData ......................................................... 13
      2.2.4.8 PPTStateData ................................................................................... 14
      2.2.4.9 ServiceError ..................................................................................... 14
      2.2.4.10 ServiceResult .................................................................................. 15
      2.2.4.11 WordStateData ............................................................................... 15
    2.2.5 Simple Types ............................................................................................... 16
      2.2.5.1 BroadcastAppCapability .................................................................. 16
      2.2.5.2 BroadcastDataKey ............................................................................ 17
      2.2.5.3 BroadcastState .................................................................................. 18
      2.2.5.4 ClientActions .................................................................................... 18
      2.2.5.5 PPTMediaPlaybackState .................................................................. 19
      2.2.5.6 PPTSlideShowState ......................................................................... 19
      2.2.5.7 ServiceErrorType ............................................................................. 20
    2.2.6 Attributes ................................................................................................... 20
    2.2.7 Groups ......................................................................................................... 20
    2.2.8 Attribute Groups ....................................................................................... 20

3 Protocol Details ................................................................................................... 21
  3.1 Server Details .................................................................................................. 21
    3.1.1 Abstract Data Model .................................................................................. 22
    3.1.2 Timers ........................................................................................................ 23
    3.1.3 Initialization ............................................................................................... 23
    3.1.4 Message Processing Events and Sequencing Rules .................................. 23
      3.1.4.1 BroadcastEndSession ....................................................................... 23
      3.1.4.1.1 Messages ....................................................................................... 24
      3.1.4.1.1.1 BroadcastEndSessionSoapIn ................................................... 24
      3.1.4.1.1.2 BroadcastEndSessionSoapOut ............................................... 24
      3.1.4.1.2 Elements ....................................................................................... 24
      3.1.4.1.2.1 BroadcastEndSession ............................................................... 24

3 / 53

[MS-OBPS] - v20200818
Office Broadcast Presentation Service
Copyright © 2020 Microsoft Corporation
Release: August 18, 2020
3.1.4.1.2.2 BroadcastEndSessionResponse ............................................ 25
3.1.4.1.3 Complex Types ........................................................................ 25
3.1.4.1.4 Simple Types ........................................................................... 25
3.1.4.1.5 Attributes ............................................................................... 25
3.1.4.1.6 Groups .................................................................................... 25
3.1.4.1.7 Attribute Groups ...................................................................... 25
3.1.4.2 BroadcastGetAppCapabilities ......................................................... 25
3.1.4.2.1 Messages ................................................................................ 26
3.1.4.2.1.1 BroadcastGetAppCapabilitiesSoapIn ..................................... 26
3.1.4.2.1.2 BroadcastGetAppCapabilitiesSoapOut .................................. 26
3.1.4.2.2 Elements ............................................................................... 26
3.1.4.2.2.1 BroadcastGetAppCapabilities ............................................. 26
3.1.4.2.2.2 BroadcastGetAppCapabilitiesResponse .................................. 27
3.1.4.2.3 Complex Types ...................................................................... 27
3.1.4.2.4 Simple Types ........................................................................... 27
3.1.4.2.5 Attributes ............................................................................... 27
3.1.4.2.6 Groups .................................................................................... 28
3.1.4.2.7 Attribute Groups ...................................................................... 28
3.1.4.3 BroadcastPutData ........................................................................ 28
3.1.4.3.1 Messages ................................................................................ 28
3.1.4.3.1.1 BroadcastPutDataSoapIn ....................................................... 28
3.1.4.3.1.2 BroadcastPutDataSoapOut .................................................... 28
3.1.4.3.2 Elements ............................................................................... 28
3.1.4.3.2.1 BroadcastPutData ................................................................. 29
3.1.4.3.2.2 BroadcastPutDataResponse ............................................... 30
3.1.4.3.3 Complex Types ...................................................................... 30
3.1.4.3.3.1 BroadcastDictionaryType .................................................... 30
3.1.4.3.3.2 ItemType ............................................................................ 30
3.1.4.3.4 Simple Types ........................................................................... 31
3.1.4.3.5 Attributes ............................................................................... 31
3.1.4.3.6 Groups .................................................................................... 31
3.1.4.3.7 Attribute Groups ...................................................................... 31
3.1.4.4 BroadcastStartSession ................................................................. 31
3.1.4.4.1 Messages ................................................................................ 31
3.1.4.4.1.1 BroadcastStartSessionSoapIn ............................................. 32
3.1.4.4.1.2 BroadcastStartSessionSoapOut ........................................... 32
3.1.4.4.2 Elements ............................................................................... 32
3.1.4.4.2.1 BroadcastStartSession ......................................................... 32
3.1.4.4.2.2 BroadcastStartSessionResponse ........................................ 32
3.1.4.4.3 Complex Types ...................................................................... 33
3.1.4.4.4 Simple Types ........................................................................... 33
3.1.4.4.4.1 AppType ............................................................................. 33
3.1.4.4.5 Attributes ............................................................................... 33
3.1.4.4.6 Groups .................................................................................... 34
3.1.4.4.7 Attribute Groups ...................................................................... 34
3.1.4.5 BroadcastPing ............................................................................. 34
3.1.4.5.1 Messages ................................................................................ 34
3.1.4.5.1.1 BroadcastPingSoapIn .......................................................... 34
3.1.4.5.1.2 BroadcastPingSoapOut ......................................................... 34
3.1.4.5.2 Elements ............................................................................... 34
3.1.4.5.2.1 BroadcastPing ................................................................. 35
3.1.4.5.2.2 BroadcastPingResponse ..................................................... 35
3.1.4.5.3 Complex Types ...................................................................... 35
3.1.4.5.4 Simple Types ........................................................................... 35
3.1.4.5.5 Attributes ............................................................................... 35
3.1.4.5.6 Groups .................................................................................... 35
3.1.4.5.7 Attribute Groups ...................................................................... 35
3.1.5 Timer Events .................................................................................. 35
3.1.6 Other Local Events

4 Protocol Examples
4.1 Presenter Client Example

5 Security
5.1 Security Considerations for Implementers
5.2 Index of Security Parameters

6 Appendix A: Full WSDL

7 Appendix B: Product Behavior

8 Change Tracking

9 Index
1 Introduction

The Office Broadcast Presentation Service Protocol enables a protocol client to create and update information about the state of a document being shared 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:

**broadcast session**: A sharing session initiated by a presenter that is used for sharing the presenter's view of a document with one or more attendees.

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

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

**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.

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

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

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

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

**time code**: A digital signal applied to a stream. The signal assigns a number to every frame of video, representing hours, minutes, seconds, and frames.

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

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

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

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

### 1.2 References

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

**1.2.1 Normative References**

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

[MS-OBPAS] Microsoft Corporation, "Office Broadcast Participant Service".


1.3 Overview

When meeting remotely, it is useful to share the current view of a document so others might follow along with the document sharer. To share this view in real-time and allow attendees to follow along with the presenter as they move through the document, information regarding the presenter’s current view of the document needs to be shared between the presenter and attendees.

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

1.4 Relationship to Other Protocols

This protocol uses the Simple Object Access Protocol (SOAP) message protocol for formatting request and response messages, as described in [SOAP1.1], [SOAP1.2-1/2007] and [SOAP1.2-2/2007]. It transmits those messages by using Hypertext Transfer Protocol (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:

![Diagram of messaging and transport stack]

Figure 1: This protocol in relation to other protocols
This protocol works in conjunction with the Office Broadcast Participant Service Protocol as specified in [MS-OBPAS] and uses the Web Application Open Platform Interface Protocol for file retrieval as specified in [MS-WOPI].

1.5 Prerequisites/Preconditions

This protocol operates between a protocol client and a protocol server that is identified by a URL that is known by protocol clients. The protocol server endpoint is formed by appending "/m/Present_2_0.asmx" to the URL of the site, for example: http://www.example.com/m/Present_2_0.asmx.

This protocol requires that files being used in broadcast sessions be identified and accessed as specified in [MS-WOPI].

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

1.6 Applicability Statement

This protocol is designed to begin and end broadcast sessions on the protocol server and to store and update broadcast session information on the protocol server.

1.7 Versioning and Capability Negotiation

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 protocol clients.

Protocol messages MUST be formatted as specified either in [SOAP1.1] section 4 or in [SOAP1.2-1/2007] 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 [SOAP1.2-1/2007] 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 [XMLSCHEMA1/2] and [XMLSCHEMA2/2], and Web Services Description Language (WSDL) as defined in [WSDL].

2.2.1 Namespaces

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace URI</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>http</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/http/">http://schemas.xmlsoap.org/wsdl/http/</a></td>
<td></td>
</tr>
<tr>
<td>soap</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/soap/">http://schemas.xmlsoap.org/wsdl/soap/</a></td>
<td>[SOAP1.1]</td>
</tr>
<tr>
<td>soap12</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/soap12/">http://schemas.xmlsoap.org/wsdl/soap12/</a></td>
<td>[SOAP1.2-1/2007] [SOAP1.2-2/2007]</td>
</tr>
<tr>
<td>wsdl</td>
<td><a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a></td>
<td>[WSDL]</td>
</tr>
<tr>
<td>xs</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>[XMLSCHEMA1/2] [XMLSCHEMA2/2]</td>
</tr>
</tbody>
</table>

2.2.2 Messages

This specification does not define any common WSDL message definitions.
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.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppServerInfo</td>
<td>A complex type that contains settings that dictate the behavior of the protocol server.</td>
</tr>
<tr>
<td>ArrayOfPPTBroadcastAnimationStepData</td>
<td>A complex type that specifies a list of PPTBroadcastAnimationStepData (section 2.2.4.6) elements. Each element in the list specifies a step in an animation timeline. Each entry in the list MUST have a unique value in its TimelineId field.</td>
</tr>
<tr>
<td>ArrayOfPPTBroadcastMediaStateData</td>
<td>A complex type that specifies a list of PPTBroadcastMediaStateData (section 2.2.4.7) elements. Each element in the list specifies a state for a multimedia object. Each entry in the list MUST have a unique value in its MediaId field.</td>
</tr>
<tr>
<td>BroadcastUser</td>
<td>A complex type that specifies a user of a broadcast session.</td>
</tr>
<tr>
<td>EnumTypes</td>
<td>Reserved. MUST be ignored.</td>
</tr>
<tr>
<td>PPTBroadcastAnimationStepData</td>
<td>A complex type that specifies a step in an animation timeline.</td>
</tr>
<tr>
<td>PPTBroadcastMediaStateData</td>
<td>A complex type that specifies a state for a multimedia object such as an embedded video clip.</td>
</tr>
<tr>
<td>PPTStateData</td>
<td>A complex type that specifies data about the state of a broadcast session of a presentation.</td>
</tr>
<tr>
<td>ServiceError</td>
<td>A complex type that specifies error information returned by the protocol server to a protocol client.</td>
</tr>
<tr>
<td>ServiceResult</td>
<td>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.</td>
</tr>
<tr>
<td>WordStateData</td>
<td>A complex type that specifies data about the state of a broadcast session of a Word document.</td>
</tr>
</tbody>
</table>

2.2.4.1 AppServerInfo


A complex type that contains settings that dictate the behavior of the protocol server.

```xml
<xs:complexType name="AppServerInfo" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="AppCapabilities" type="tns:BroadcastDictionaryType"/>
  </xs:sequence>
</xs:complexType>
```
AppCapabilities: A BroadcastDictionaryType (section 3.1.4.3.1) element that contains server-related settings. This element MUST be present.

2.2.4.2 ArrayOfPPTBroadcastAnimationStepData


A complex type that specifies a list of PPTBroadcastAnimationStepData (section 2.2.4.6) elements. Each element in the list specifies a step in an animation timeline. Each entry in the list MUST have a unique value in its TimelineId field.

ArrayOfPPTBroadcastAnimationStepData: Specifies a PPTBroadcastAnimationStepData (section 2.2.4.6) element. The element MUST be present.

2.2.4.3 ArrayOfPPTBroadcastMediaStateData


A complex type that specifies a list of PPTBroadcastMediaStateData (section 2.2.4.7) elements. Each element in the list specifies a state for a multimedia object. Each entry in the list MUST have a unique value in its MediaId field.

ArrayOfPPTBroadcastMediaStateData: Each element specifies a PPTBroadcastMediaStateData (section 2.2.4.7) element. The element MUST be present.

2.2.4.4 BroadcastUser


A complex type that specifies a user of a broadcast session.
**SessionId:** An `xs:string` ([XMLSCHEMA2/2] 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` ([XMLSCHEMA2/2] 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.

### 2.2.4.5 EnumTypes

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

Reserved. MUST be ignored.

```xml
<xs:complexType name="EnumTypes" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="BroadcastDataKey" type="tns:BroadcastDataKey"/>
    <xs:element minOccurs="1" maxOccurs="1" name="BroadcastAppCapability" type="tns:BroadcastAppCapability"/>
    <xs:element minOccurs="1" maxOccurs="1" name="AppType" type="tns:AppType"/>
    <xs:element minOccurs="1" maxOccurs="1" name="BroadcastState" type="tns:BroadcastState"/>
    <xs:element minOccurs="1" maxOccurs="1" name="PPTSlideShowState" type="tns:PPTSlideShowState"/>
    <xs:element minOccurs="1" maxOccurs="1" name="PPTMediaPlaybackState" type="tns:PPTMediaPlaybackState"/>
  </xs:sequence>
</xs:complexType>
```

**BroadcastDataKey:** This element is reserved and MUST be ignored.

**BroadcastAppCapability:** This element is reserved and MUST be ignored.

**AppType:** This element is reserved and MUST be ignored.

**BroadcastState:** This element is reserved and MUST be ignored.

**PPTSlideShowState:** This element is reserved and MUST be ignored.

**PPTMediaPlaybackState:** This element is reserved and MUST be ignored.

### 2.2.4.6 PPTBroadcastAnimationStepData

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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

```xml
<xs:complexType name="PPTBroadcastAnimationStepData" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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` ([XMLSCHEMA2/2] section 3.2.1) element that specifies an identifier of the animation timeline. This element MUST be present.

**Step:** An `xs:int` ([XMLSCHEMA2/2] section 3.3.17) element that specifies the step number in the animation timeline given by the `TimelineId` field. This element MUST be present.
2.2.4.7 PPTBroadcastMediaStateData


A complex type that specifies a state for a multimedia object such as an embedded video clip.

```xml
<xs:complexType name="PPTBroadcastMediaStateData" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="MediaId" type="xs:string"/>
    <xs:element minOccurs="1" maxOccurs="1" name="State" type="tns:PPTMediaPlaybackState"/>
    <xs:element minOccurs="1" maxOccurs="1" name="AtPosition" type="xs:double"/>
  </xs:sequence>
</xs:complexType>
```

**MediaId:** An `xs:string` ([XMLSCHEMA2/2] section 3.2.1) element that specifies an identifier of the multimedia object. This element MUST be present.

**State:** A `PPTMediaPlaybackState` (section 2.2.5.5) element that specifies the playback state of the multimedia object identified by the `MediaId` field. This element MUST be present.

**AtPosition:** An `xs:double` ([XMLSCHEMA2/2] section 3.2.5) element that specifies the time code of the multimedia object identified by the `MediaId` field when the `State` field is changed. This element MUST be present.

2.2.4.8 PPTStateData


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

```xml
<xs:complexType name="PPTStateData" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="SlideId" type="xs:unsignedInt"/>
    <xs:element minOccurs="1" maxOccurs="1" name="SlideIndex" type="xs:unsignedInt"/>
    <xs:element minOccurs="0" maxOccurs="1" name="AnimationStepDataList" type="tns:ArrayOfPPTBroadcastAnimationStepData"/>
    <xs:element minOccurs="0" maxOccurs="1" name="MediaStateDataList" type="tns:ArrayOfPPTBroadcastMediaStateData"/>
    <xs:element minOccurs="1" maxOccurs="1" name="PPTSlideShowState" type="tns:PPTSlideShowState"/>
  </xs:sequence>
</xs:complexType>
```

**SlideId:** An `xs:unsignedInt` ([XMLSCHEMA2/2] section 3.3.22) element that specifies the identifier of the presentation slide. This element MUST be present.

**SlideIndex:** An `xs:unsignedInt` ([XMLSCHEMA2/2] section 3.3.22) element that specifies the zero-based ordered index of the presentation slide. This element MUST be present if `SlideId` is equal to 0. This element MUST be ignored if `SlideId` is not equal to 0.

**AnimationStepDataList:** An `ArrayOfPPTBroadcastAnimationStepData` (section 2.2.4.2) element that specifies the state of each of the animation timelines. This element MUST be present.

**MediaStateDataList:** An `ArrayOfPPTBroadcastMediaStateData` (section 2.2.4.3) element that specifies the state of each of the multimedia objects. This element MUST be present.

**PPTSlideShowState:** A `PPTSlideShowState` (section 2.2.5.6) element that specifies the current state of the slide show. This element MUST be present.
2.2.4.9  ServiceError

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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

```xml
<xs:complexType name="ServiceError" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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` ([XMLSCHEMA2/2] 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` ([XMLSCHEMA2/2] 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** (section 2.2.5.7) element that specifies the error type. This element MUST be present.

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

2.2.4.10  ServiceResult

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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.

```xml
<xs:complexType name="ServiceResult" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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` ([XMLSCHEMA1/2] 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** (section 2.2.4.9) element that specifies an error result of a protocol message response. This element MUST NOT be present if the **Result** element is present.

2.2.4.11  WordStateData

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

A complex type that specifies data about the state of a broadcast session of a Word document.

```xml
<xs:complexType name="WordStateData" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
  </xs:sequence>
</xs:complexType>
```
Page: An `xs:int` ([XMLSCHEMA2/2] section 3.3.17) element that specifies the one-based ordered index of the current page within the set of all pages of the document. This element MUST be present.

Offset: An `xs:double` ([XMLSCHEMA2/2] section 3.2.5) element that specifies a percentage offset from the top of the page defined by the Page field. This element MUST be 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.

<table>
<thead>
<tr>
<th>Simple type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastAppCapability</td>
<td>A simple type that specifies an enumeration of a set of capabilities returned by the protocol server to the protocol client.</td>
</tr>
<tr>
<td>BroadcastDataKey</td>
<td>A simple type that specifies an enumeration of a set of keys to data stored in the protocol server for a given broadcast session.</td>
</tr>
<tr>
<td>BroadcastState</td>
<td>A simple type that specifies an enumeration of all the possible broadcast session states.</td>
</tr>
<tr>
<td>ClientActions</td>
<td>A simple type that is reserved and MUST be ignored.</td>
</tr>
<tr>
<td>PPTMediaPlaybackState</td>
<td>A simple type that specifies an enumeration of the different states a multimedia object can be in.</td>
</tr>
<tr>
<td>PPTSlideShowState</td>
<td>A simple type that specifies an enumeration of all the possible slide show states.</td>
</tr>
<tr>
<td>ServiceErrorType</td>
<td>A simple type that specifies an enumeration of a set of protocol errors returned by the protocol server to the protocol client.</td>
</tr>
</tbody>
</table>

#### 2.2.5.1 BroadcastAppCapability

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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

```xml
<x:simpleType name="BroadcastAppCapability" xmlns:x="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SessionTimeout"/>
    <xs:enumeration value="SessionIdleTimeOut"/>
    <xs:enumeration value="SupportVideo"/>
    <xs:enumeration value="SupportAudio"/>
    <xs:enumeration value="SupportNotes"/>
    <xs:enumeration value="MediaExtensions"/>
    <xs:enumeration value="MaxMediaSize"/>
  </xs:restriction>
</xs:simpleType>
```
The following table specifies the allowable values for the **BroadcastAppCapability** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessionTimeout</td>
<td>This capability specifies the maximum duration in seconds for a broadcast session after which the session is ended.</td>
</tr>
<tr>
<td>SessionIdleTimeOut</td>
<td>This capability specifies the maximum duration in seconds after which a broadcast session is ended if there has been no presenter input.</td>
</tr>
<tr>
<td>SupportVideo</td>
<td>This capability specifies whether the protocol server supports video content in the broadcast session.</td>
</tr>
<tr>
<td>SupportAudio</td>
<td>This capability specifies whether the protocol server supports audio content in the broadcast session.</td>
</tr>
<tr>
<td>SupportNotes</td>
<td>This capability specifies whether the protocol server supports note-taking through OneNote in the broadcast session.</td>
</tr>
<tr>
<td>MediaExtensions</td>
<td>This capability specifies the supported file name extensions for multimedia content.</td>
</tr>
<tr>
<td>MaxMediaSize</td>
<td>This capability specifies the maximum size in bytes of an individual multimedia object that is rendered on the protocol server.</td>
</tr>
</tbody>
</table>

### 2.2.5.2 BroadcastDataKey

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

A simple type that specifies an enumeration of a set of keys to data stored in the protocol server for a given broadcast session.

```xml
<xs:simpleType name="BroadcastDataKey" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="AppType"/>
    <xs:enumeration value="SequenceNumber"/>
    <xs:enumeration value="FileVersion"/>
    <xs:enumeration value="OriginalFileName"/>
    <xs:enumeration value="BroadcastState"/>
    <xs:enumeration value="AppSpecificStateData"/>
    <xs:enumeration value="NotesUrl"/>
    <xs:enumeration value="DataVersion"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for the **BroadcastDataKey** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppType</td>
<td>This key is used to specify an application type in the key field of an ItemType (section 3.1.4.3.3.2).</td>
</tr>
<tr>
<td>SequenceNumber</td>
<td>This key is used to specify a monotonically increasing sequence number in the key field of an ItemType (section 3.1.4.3.3.2).</td>
</tr>
<tr>
<td>FileVersion</td>
<td>This key is used to specify a monotonically increasing file version number in the key field of an ItemType (section 3.1.4.3.3.2).</td>
</tr>
<tr>
<td>OriginalFileName</td>
<td>This key is used to specify the file name of the document in the key field of an ItemType (section 3.1.4.3.3.2).</td>
</tr>
</tbody>
</table>
### BroadcastState


A simple type that specifies an enumeration of all the possible broadcast session states.

```xml
<xs:simpleType name="BroadcastState" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="BroadcastNotStartedYet"/>
    <xs:enumeration value="BroadcastStarted"/>
    <xs:enumeration value="BroadcastEnded"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for the BroadcastState simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastNotStartedYet</td>
<td>The broadcast session has not started yet.</td>
</tr>
<tr>
<td>BroadcastStarted</td>
<td>The broadcast session has started but not ended yet.</td>
</tr>
<tr>
<td>BroadcastEnded</td>
<td>The broadcast session has ended.</td>
</tr>
</tbody>
</table>

### ClientActions


A simple type that is reserved and MUST be ignored.

```xml
<xs:simpleType name="ClientActions" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:list>
    <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:list>
</xs:simpleType>
```
The following table specifies the allowable values for the **ClientActions** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Reserved and MUST be ignored.</td>
</tr>
<tr>
<td>Dismiss</td>
<td>Reserved and MUST be ignored.</td>
</tr>
<tr>
<td>Close</td>
<td>Reserved and MUST be ignored.</td>
</tr>
<tr>
<td>OpenInClient</td>
<td>Reserved and MUST be ignored.</td>
</tr>
<tr>
<td>Refresh</td>
<td>Reserved and MUST be ignored.</td>
</tr>
</tbody>
</table>

### 2.2.5.5 PPTMediaPlaybackState

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

A simple type that specifies an enumeration of the different states a multimedia object can be in.

```xml
<xs:simpleType name="PPTMediaPlaybackState" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Paused"/>
    <xs:enumeration value="Playing"/>
    <xs:enumeration value="Stopped"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for the **PPTMediaPlaybackState** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paused</td>
<td>The multimedia object is in a paused state.</td>
</tr>
<tr>
<td>Playing</td>
<td>The multimedia object is in a playing state.</td>
</tr>
<tr>
<td>Stopped</td>
<td>The multimedia object is in a stopped state.</td>
</tr>
</tbody>
</table>

### 2.2.5.6 PPTSlideShowState

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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

```xml
<xs:simpleType name="PPTSlideShowState" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="BlackScreen"/>
    <xs:enumeration value="WhiteScreen"/>
    <xs:enumeration value="Normal"/>
    <xs:enumeration value="SlideShowEnded"/>
  </xs:restriction>
</xs:simpleType>
```
The following table specifies the allowable values for the **PPTSlideShowState** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlackScreen</td>
<td>The slide show is displaying a black screen.</td>
</tr>
<tr>
<td>WhiteScreen</td>
<td>The slide show is displaying a white screen.</td>
</tr>
<tr>
<td>Normal</td>
<td>The slide show is displaying presentation slides.</td>
</tr>
<tr>
<td>SlideShowEnded</td>
<td>The slide show has ended.</td>
</tr>
</tbody>
</table>

### 2.2.5.7 ServiceErrorType

**Namespace:** http://schemas.microsoft.com/server/broadcast/2010/main

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

```xml
<xs:simpleType name="ServiceErrorType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="UnknownError"/>
    <xs:enumeration value="ApplicationError"/>
    <xs:enumeration value="Timeout"/>
    <xs:enumeration value="ServiceBusy"/>
    <xs:enumeration value="SessionFull"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for the **ServiceErrorType** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnknownError</td>
<td>The protocol server encountered an unknown error.</td>
</tr>
<tr>
<td>ApplicationError</td>
<td>The protocol server encountered an application error.</td>
</tr>
<tr>
<td>Timeout</td>
<td>The protocol server encountered an application timeout.</td>
</tr>
<tr>
<td>ServiceBusy</td>
<td>The protocol server is busy.</td>
</tr>
<tr>
<td>SessionFull</td>
<td>The protocol server cannot allow more attendees to join a broadcast session.</td>
</tr>
</tbody>
</table>

### 2.2.6 Attributes

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

### 2.2.7 Groups

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

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

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

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

Except where specified, protocol clients SHOULD interpret Hypertext Transfer Protocol (HTTP) status codes that are returned by the protocol server as specified in [RFC2616], section 10.

This protocol allows protocol servers to notify protocol clients of application-level faults by using SOAP faults. Except where otherwise 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 to notify protocol clients of authorization faults by using either HTTP status codes or SOAP faults, as specified previously in this section.

3.1 Server Details

The following high-level sequence diagram illustrates the operation of the presenter client protocol.
First, a protocol client acting as broadcast presenter sends a BroadcastGetAppCapabilities (section 3.1.4.2) message. The server responds with a BroadcastGetAppCapabilitiesResponse message that contains the capabilities supported by the server. Next, the protocol client sends a BroadcastStartSession (section 3.1.4.4) message and the protocol server responds with a BroadcastStartSessionResponse message indicating the result of the operation. Next, the protocol client sends one or more BroadcastPutData (section 3.1.4.3) messages containing the current state of the broadcast, and the server sends a BroadcastPutDataResponse message to acknowledge the request. When the broadcast is finished, the protocol client then sends a BroadcastEndSession (section 3.1.4.1) 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.

Note that the abstract interface notation "(Public)" indicates that the abstract data model element can be directly accessed from outside this protocol.
SessionId (Public): An entity that represents a unique identifier for a broadcast session.

BroadcastState (Public): An entity that represents the state of the broadcast session.

AppSpecificStateData (Public): An entity that represents state specific to the application for which this protocol is being used.

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.

The following table summarizes the list of operations as defined by this specification.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastEndSession</td>
<td>The BroadcastEndSession operation is used by the protocol client to end a broadcast session on the protocol server.</td>
</tr>
<tr>
<td>BroadcastGetAppCapabilities</td>
<td>The BroadcastGetAppCapabilities operation is used by the protocol client to retrieve session-independent settings from the protocol server.</td>
</tr>
<tr>
<td>BroadcastPing</td>
<td>The BroadcastPing operation is used by a protocol client to check if the protocol server is available.</td>
</tr>
<tr>
<td>BroadcastPutData</td>
<td>The BroadcastPutData operation is used by the protocol client to modify the current state of the broadcast session on the protocol server.</td>
</tr>
<tr>
<td>BroadcastStartSession</td>
<td>The BroadcastStartSession operation is used by the protocol client to begin a broadcast session on the protocol server.</td>
</tr>
</tbody>
</table>

3.1.4.1 BroadcastEndSession

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

The following is the WSDL port type specification of the BroadcastEndSession WSDL operation.

```xml
<wsdl:operation name="BroadcastEndSession" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <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, and the protocol server MUST respond with a BroadcastEndSessionSoapOut response message, as specified in section 3.1.4.1.2.
3.1.4.1.1 Messages
The following table summarizes the set of WSDL message definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastEndSessionSoapIn</td>
<td>The request WSDL message for the BroadcastEndSession WSDL operation.</td>
</tr>
<tr>
<td>BroadcastEndSessionSoapOut</td>
<td>The response WSDL message for the BroadcastEndSession WSDL operation.</td>
</tr>
</tbody>
</table>

3.1.4.1.1.1 BroadcastEndSessionSoapIn
The request WSDL message for the BroadcastEndSession WSDL operation.
The SOAP action value is:


The SOAP body contains the BroadcastEndSession element.

3.1.4.1.1.2 BroadcastEndSessionSoapOut
The response WSDL message for the BroadcastEndSession WSDL operation.
The SOAP body contains the BroadcastEndSessionResponse element.

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

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastEndSession</td>
<td>The input data for the BroadcastEndSession WSDL operation.</td>
</tr>
<tr>
<td>BroadcastEndSessionResponse</td>
<td>The result data for the BroadcastEndSession WSDL operation.</td>
</tr>
</tbody>
</table>

3.1.4.1.2.1 BroadcastEndSession
The BroadcastEndSession element specifies the input data for the BroadcastEndSession WSDL operation.

    <xs:element name="BroadcastEndSession" xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <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 (section 2.2.4.4) that is obtained through a BroadcastStartSession operation (section 3.1.4.4). This element MUST be present.

3.1.4.1.2.2 BroadcastEndSessionResponse

The BroadcastEndSessionResponse element specifies the result data for the BroadcastEndSession WSDL operation.

```xml
<xs:element name="BroadcastEndSessionResponse" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastEndSessionResult" type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastEndSessionResult: A ServiceResult (section 2.2.4.10) 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.

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 BroadcastGetAppCapabilities

The BroadcastGetAppCapabilities operation is used by the protocol client to retrieve session-independent settings from the protocol server.

The following is the WSDL port type specification of the BroadcastGetAppCapabilities WSDL operation.

```xml
<wSDL:operation name="BroadcastGetAppCapabilities" xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/">
  <wSDL:input message="tns:BroadcastGetAppCapabilitiesSoapIn"/>
  <wSDL:output message="tns:BroadcastGetAppCapabilitiesSoapOut"/>
</wSDL:operation>
```
The protocol client sends a **BroadcastGetAppCapabilitiesSoapIn** request message, as specified in section 3.1.4.2.1, and the protocol server MUST respond with a **BroadcastGetAppCapabilitiesSoapOut** response message, as specified in section 3.1.4.2.2.

### 3.1.4.2.1 Messages

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

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastGetAppCapabilitiesSoapIn</td>
<td>The request WSDL message for the <strong>BroadcastGetAppCapabilities WSDL operation</strong>.</td>
</tr>
<tr>
<td>BroadcastGetAppCapabilitiesSoapOut</td>
<td>The response WSDL message for the <strong>BroadcastGetAppCapabilities WSDL operation</strong>.</td>
</tr>
</tbody>
</table>

#### 3.1.4.2.1.1 **BroadcastGetAppCapabilitiesSoapIn**

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

The *SOAP action* value is:

```
```

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

#### 3.1.4.2.1.2 **BroadcastGetAppCapabilitiesSoapOut**

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

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

### 3.1.4.2.2 Elements

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

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastGetAppCapabilities</td>
<td>The input data for the <strong>BroadcastGetAppCapabilities WSDL operation</strong>.</td>
</tr>
<tr>
<td>BroadcastGetAppCapabilitiesResponse</td>
<td>The result data for the <strong>BroadcastGetAppCapabilities WSDL operation</strong>.</td>
</tr>
</tbody>
</table>

#### 3.1.4.2.2.1 **BroadcastGetAppCapabilities**

The **BroadcastGetAppCapabilities** element specifies the input data for the **BroadcastGetAppCapabilities WSDL operation**.

```xml
<xs:element name="BroadcastGetAppCapabilities" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType/>
```

---

*Office Broadcast Presentation Service*

Copyright © 2020 Microsoft Corporation

Release: August 18, 2020
### 3.1.4.2.2.2 BroadcastGetAppCapabilitiesResponse

The **BroadcastGetAppCapabilitiesResponse** element specifies the result data for the **BroadcastGetAppCapabilities** WSDL operation.

```xml
<xs:element name="BroadcastGetAppCapabilitiesResponse" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastGetAppCapabilitiesResult" type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**BroadcastGetAppCapabilitiesResult:** A **ServiceResult** (section 2.2.4.10) that specifies the result of the operation. This element MUST be present. If successful, the **Result** child element of the **ServiceResult** MUST be an **AppServerInfo** (section 2.2.4.1) element. Otherwise, the **Error** child element of the **ServiceResult** MUST be present.

The **AppServerInfo** contains a **BroadcastDictionaryType** (section 3.1.4.3.3.1), which contains a list of **ItemType** (section 3.1.4.3.3.2) elements.

The **key** field in an **ItemType** element MUST be a **BroadcastAppCapability** (section 2.2.5.1) value of type **xs:string** ([XMLSCHEMA2/2] section 3.2.1).

The **value** field of an **ItemType** element MUST be an **xs:string** ([XMLSCHEMA2/2] section 3.2.1) element, as specified in the following table.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessionTimeout</td>
<td>MUST be a non-negative integer</td>
</tr>
<tr>
<td>SessionIdleTimeOut</td>
<td>MUST be a non-negative integer</td>
</tr>
<tr>
<td>SupportVideo</td>
<td>MUST be &quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>SupportAudio</td>
<td>MUST be &quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>SupportNotes</td>
<td>MUST be &quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>MediaExtensions</td>
<td>MUST be a string containing a comma-separated list of file name extensions</td>
</tr>
<tr>
<td>MaxMediaSize</td>
<td>MUST be a non-negative integer</td>
</tr>
</tbody>
</table>

### 3.1.4.2.3 Complex Types

None.

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

The following is the WSDL port type specification of the BroadcastPutData WSDL operation.

```xml
<wSDL:operation name="BroadcastPutData" xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/">
  <wSDL:input message="tns:BroadcastPutDataSoapIn"/>
  <wSDL:output message="tns:BroadcastPutDataSoapOut"/>
</wSDL:operation>
```

The protocol client sends a BroadcastPutDataSoapIn request message, as specified in section 3.1.4.3.1.1, and the protocol server MUST respond with a BroadcastPutDataSoapOut response message, as specified in section 3.1.4.3.1.2.

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

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastPutDataSoapIn</td>
<td>The request WSDL message for the BroadcastPutData WSDL operation.</td>
</tr>
<tr>
<td>BroadcastPutDataSoapOut</td>
<td>The response WSDL message for the BroadcastPutData WSDL operation.</td>
</tr>
</tbody>
</table>

3.1.4.3.1.1 BroadcastPutDataSoapIn
The request WSDL message for the BroadcastPutData WSDL operation.

The SOAP action value is:


The SOAP body contains the BroadcastPutData element.

3.1.4.3.1.2 BroadcastPutDataSoapOut
The response WSDL message for the BroadcastPutData WSDL operation.

The SOAP body contains the BroadcastPutDataResponse element.

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

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastPutData</td>
<td>The input data for the BroadcastPutData WSDL operation.</td>
</tr>
<tr>
<td>BroadcastPutDataResponse</td>
<td>The result data for the BroadcastPutData WSDL operation.</td>
</tr>
</tbody>
</table>

### 3.1.4.3.2.1 BroadcastPutData

The **BroadcastPutData** element specifies the input data for the **BroadcastPutData** WSDL operation.

```xml
<xs:element name="BroadcastPutData" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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:BroadcastDictionaryType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**user**: A **BroadcastUser** (section 2.2.4.4) element that is obtained by making a **BroadcastStartSession** (section 3.1.4.4) web method call. This element MUST be present.

**data**: A **BroadcastDictionaryType** (section 3.1.4.3.3.1) element that specifies the current broadcast session on the protocol client. This element MUST be present.

The **BroadcastDictionaryType** contains a list of **ItemType** (section 3.1.4.3.3.2) elements. The **key** in each **ItemType** element MUST be a **BroadcastDataKey** (section 2.2.5.2) value of type xs:string. The value field in an **ItemType** element MUST conform to the following table.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppType</td>
<td>MUST be an <strong>AppType</strong> (section 3.1.4.4.4.1) value of type xs:string ([XMLSCHEMA2/2] section 3.2.1).</td>
</tr>
<tr>
<td>SequenceNumber</td>
<td>MUST be an xs:string [XMLSCHEMA2/2] section 3.2.1 element. The numeric value MUST follow an xs:int ([XMLSCHEMA2/2] section 3.3.17) element.</td>
</tr>
<tr>
<td>OriginalFileName</td>
<td>MUST be an xs:string ([XMLSCHEMA2/2] section 3.2.1) element.</td>
</tr>
<tr>
<td>BroadcastState</td>
<td>MUST be a <strong>BroadcastState</strong> (section 2.2.5.3) element.</td>
</tr>
<tr>
<td>AppSpecificStateData</td>
<td>MUST be an xs:string [XMLSCHEMA2/2] section 3.2.1 element. The <strong>AppSpecificStateData</strong> MUST have the following layout: MUST be a JavaScript Object Notation (JSON) [RFC4627] serialization of a PPTStateData (section 2.2.4.8) element if the <strong>AppType</strong> is PPT, or a JSON [RFC4627] serialization of a <strong>WordStateData</strong> (section 2.2.4.11) element if the <strong>AppType</strong> is Word.</td>
</tr>
<tr>
<td>NotesUrl</td>
<td>MUST be an xs:string ([XMLSCHEMA2/2] section 3.2.1) element.</td>
</tr>
</tbody>
</table>
### Key

<table>
<thead>
<tr>
<th><strong>Key</strong></th>
<th><strong>Value</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>DataVersion</td>
<td>MUST be an <code>xs:string</code> ([XMLSCHEMA2/2] section 3.3.17) element with a value of &quot;2&quot;.</td>
</tr>
</tbody>
</table>

### 3.1.4.3.2.2 BroadcastPutDataResponse

The **BroadcastPutDataResponse** element specifies the result data for the **BroadcastPutData WSDL operation**.

```xml
<xs:element name="BroadcastPutDataResponse" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastPutDataResult" type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**BroadcastPutDataResult**: A **ServiceResult** (section 2.2.4.10) 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.

### 3.1.4.3.3 Complex Types

The following table summarizes the **XML schema** complex type definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastDictionaryType</td>
<td>A complex type that specifies a list of <strong>ItemType</strong> (section 3.1.4.3.3.2) elements.</td>
</tr>
<tr>
<td>ItemType</td>
<td>A complex type that holds a key-value pair.</td>
</tr>
</tbody>
</table>

#### 3.1.4.3.3.1 BroadcastDictionaryType

**Namespace**: http://schemas.microsoft.com/server/broadcast/2010/main

A complex type that specifies a list of **ItemType** (section 3.1.4.3.3.2) elements.

```xml
<xs:complexType name="BroadcastDictionaryType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="item" type="tns:ItemType"/>
  </xs:sequence>
</xs:complexType>
```

**item**: Each element specifies an **ItemType** (section 3.1.4.3.3.2) element. The element MUST be present.

#### 3.1.4.3.3.2 ItemType

**Namespace**: http://schemas.microsoft.com/server/broadcast/2010/main

A complex type that holds a key-value pair.
key: An `xs:string` ([XMLSCHEMA2/2] section 3.2.1) element that specifies the key in a key-value pair. This element MUST be present.

value: An `xs:string` ([XMLSCHEMA2/2] section 3.2.1) element that specifies the value in a key-value pair. This element MUST be present.

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 BroadcastStartSession

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

The following is the WSDL port type specification of the `BroadcastStartSession` WSDL operation.

```xml
<wsdl:operation name="BroadcastStartSession" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:input message="tns:BroadcastStartSessionSoapIn"/>
  <wsdl:output message="tns:BroadcastStartSessionSoapOut"/>
</wsdl:operation>
```

The protocol client sends a `BroadcastStartSessionSoapIn` request message, as specified in section 3.1.4.4.1.1, and the protocol server MUST respond with a `BroadcastStartSessionSoapOut` response message, as specified in section 3.1.4.4.1.2.

3.1.4.4.1 Messages

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

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>BroadcastStartSessionSoapIn</code></td>
<td>The request WSDL message for the <code>BroadcastStartSession</code> WSDL operation.</td>
</tr>
<tr>
<td><code>BroadcastStartSessionSoapOut</code></td>
<td>The response WSDL message for the <code>BroadcastStartSession</code> WSDL operation.</td>
</tr>
</tbody>
</table>

[MS-OBPRS] - v20200818
Office Broadcast Presentation Service
Copyright © 2020 Microsoft Corporation
Release: August 18, 2020
3.1.4.4.1.1 BroadcastStartSessionSoapIn

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

The **SOAP action** value is:


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

3.1.4.4.1.2 BroadcastStartSessionSoapOut

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

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

3.1.4.4.2 Elements

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

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastStartSession</td>
<td>The input data for the <strong>BroadcastStartSession</strong> <strong>WSDL</strong> operation.</td>
</tr>
<tr>
<td>BroadcastStartSessionResponse</td>
<td>The result data for the <strong>BroadcastStartSession</strong> <strong>WSDL</strong> operation.</td>
</tr>
</tbody>
</table>

3.1.4.4.2.1 BroadcastStartSession

The **BroadcastStartSession** element specifies the input data for the **BroadcastStartSession** **WSDL** operation.

```xml
<xs:element name="BroadcastStartSession" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="query" type="xs:string"/>
      <xs:element minOccurs="1" maxOccurs="1" name="appType" type="tns:AppType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**query**: An **xs:string** element (see [XMLSCHEMA2/2] section 3.2.1) that specifies an identifier for the broadcast session. This identifier MUST conform to the pattern **WOPISrc=<source>&access_token=<token>**, where **<source>** is the **WOPISrc** parameter as defined in [MS-WOPI] section 3.1.5.1.1.2.3.3, and **<token>** is the **<token>** parameter as defined in [MS-WOPI] section 2.2.3.

**appType**: An **AppType** (section 3.1.4.4.1) element that specifies the type of the broadcast session.

3.1.4.4.2.2 BroadcastStartSessionResponse
The **BroadcastStartSessionResponse** element specifies the result data for the **BroadcastStartSession** WSDL operation.

```xml
<xs:element name="BroadcastStartSessionResponse" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="BroadcastStartSessionResult" type="tns:ServiceResult"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**BroadcastStartSessionResult**: A **ServiceResult** (section 2.2.4.10) that specifies the result of the operation. This element MUST be present. The **Result** child element MUST be a **BroadcastUser** (section 2.2.4.4) if the **Error** child element is not present.

### 3.1.4.4.3 Complex Types

None.

### 3.1.4.4.4 Simple Types

The following table summarizes the **XML schema** simple type definitions that are specific to this operation.

<table>
<thead>
<tr>
<th>Simple type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppType</td>
<td>A simple type that specifies an enumeration of all the possible types for a broadcast session.</td>
</tr>
</tbody>
</table>

#### 3.1.4.4.4.1 AppType

**Namespace**: http://schemas.microsoft.com/server/broadcast/2010/main

A simple type that specifies an enumeration of all the possible types for a broadcast session.

```xml
<xs:simpleType name="AppType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="PPT"/>
    <xs:enumeration value="Word"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for the **AppType** simple type.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPT</td>
<td>The broadcast session is of a PowerPoint presentation.</td>
</tr>
<tr>
<td>Word</td>
<td>The broadcast session is of a Word document.</td>
</tr>
</tbody>
</table>

#### 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 BroadcastPing

The **BroadcastPing** operation is used by a protocol client to check if the protocol server is available.

The following is the **WSDL** port type specification of the **BroadcastPing WSDL operation**.

```xml
<wsdl:operation name="BroadcastPing" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:input message="tns:BroadcastPingSoapIn"/>
  <wsdl:output message="tns:BroadcastPingSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **BroadcastPingSoapIn** request message, as specified in section 3.1.4.5.1.1, and the protocol server MUST respond with a **BroadcastPingSoapOut** response message, as specified in section 3.1.4.5.1.2.

3.1.4.5.1 Messages

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

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadcastPingSoapIn</td>
<td>The request WSDL message for the <strong>BroadcastPing WSDL operation</strong>.</td>
</tr>
<tr>
<td>BroadcastPingSoapOut</td>
<td>The response WSDL message for the <strong>BroadcastPing WSDL operation</strong>.</td>
</tr>
</tbody>
</table>

3.1.4.5.1.1 BroadcastPingSoapIn

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

The **SOAP action** value is:

```xml
```

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

3.1.4.5.1.2 BroadcastPingSoapOut

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

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

3.1.4.5.2 Elements

The following table summarizes the **XML schema** element definitions that are specific to this operation.
Element | Description
--- | ---
BroadcastPing | The input data for the BroadcastPing WSDL operation.
BroadcastPingResponse | The result data for the BroadcastPing WSDL operation.

3.1.4.5.2.1 BroadcastPing

The BroadcastPing element specifies the input data for the BroadcastPing WSDL operation.

```xml
<xs:element name="BroadcastPing" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType/>
</xs:element>
```

3.1.4.5.2.2 BroadcastPingResponse

The BroadcastPingResponse element specifies the result data for the BroadcastPing WSDL operation.

```xml
<xs:element name="BroadcastPingResponse" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="BroadcastPingResult" type="xs:boolean"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BroadcastPingResult: An xs:boolean ([XMLSCHEMA2/2] section 3.2.2) element. This element MUST be present and MUST be set to true.

3.1.4.5.3 Complex Types

None.

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.
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 fetch the settings stored on the protocol server. The following **BroadcastGetAppCapabilitiesSoapIn** 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"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
  </soap:Body>
</soap:Envelope>
```

The protocol server responds with a message to acknowledge the request and to provide the protocol client with the protocol server settings and their values. The following **BroadcastGetAppCapabilitiesSoapOut** 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>
      <BroadcastGetAppCapabilitiesResult>
        <Result xsi:type="AppServerInfo">
          <AppCapabilities>
            <Item>
              <key>SessionTimeout</key>
              <value>43200</value>
            </Item>
            <Item>
              <key>SupportNotes</key>
              <value>true</value>
            </Item>
            <Item>
              <key>MediaExtensions</key>
              <value>3g2, 3gp, 3gpp, 3gp2, aac, adts, asf, asx, flv, m4a, m4b, m4v, mov, mp3, mp4, mpa, swf, wax, wm, wma, wmv, wmx, wpl, wvx</value>
            </Item>
            <Item>
              <key>MaxMediaSize</key>
              <value>10485760</value>
            </Item>
          </AppCapabilities>
        </Result>
      </BroadcastGetAppCapabilitiesResult>
    </BroadcastGetAppCapabilitiesResponse>
  </soap:Body>
</soap:Envelope>
```

The presenter protocol client then sends a request to the protocol server to begin the 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"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
  </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"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <BroadcastStartSessionResponse
      <BroadcastStartSessionResult>
        <Result xsi:type="BroadcastUser">
          <SessionId>WOPISrc=http://machinename/th/handler/wopi/files/anonymous~PPTTest.pptx&amp;access_token=VAR</SessionId>
          <UserToken>e59840b8-d523-4477-b978-5369a237775a</UserToken>
        </Result>
      </BroadcastStartSessionResult>
    </BroadcastStartSessionResponse>
  </soap:Body>
</soap:Envelope>
```

Next, the presenter protocol client sends information about the current state of the 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"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
      <user>
        <SessionId>WOPISrc=http://machinename/th/handler/wopi/files/anonymous~PPTTest.pptx&amp;access_token=VAR</SessionId>
        <UserToken>e59840b8-d523-4477-b978-5369a237775a</UserToken>
      </user>
      <data>
        <item>
          <key>AppType</key>
          <value>PPT</value>
        </item>
        <item>
          <key>BroadcastState</key>
          <value>BroadcastStarted</value>
        </item>
        <item>
          <key>SequenceNumber</key>
          <value>5</value>
        </item>
        <item>
          <key>FileVersion</key>
          <value>1</value>
        </item>
      </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:

```xml
<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>
    </soap:Body>
</soap:Envelope>
```

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

```xml
<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>
            <user>
                <SessionId>WOPISrc=http://mac
                <UserToken>e59840b8-d523-4477-b978-5369a237775a</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:

```xml
<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>
    </soap:Body>
</soap:Envelope>
```
<soap:Body>
  <BroadcastEndSessionResponse
    <BroadcastEndSessionResult />
  </BroadcastEndSessionResponse>
</soap:Body>
</soap:Envelope>
5 Security

5.1 Security Considerations for Implementers
None.

5.2 Index of Security Parameters
None.
6 Appendix A: Full WSDL

For ease of implementation, the full WSDL is provided in this appendix.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <xs:schema elementFormDefault="qualified"
      <xs:element name="BroadcastStartSession">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="query" type="xs:string"/>
            <xs:element minOccurs="1" maxOccurs="1" name="appType" type="tns:AppType"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:simpleType name="AppType">
        <xs:restriction base="xs:string">
          <xs:enumeration value="PPT"/>
          <xs:enumeration value="Word"/>
        </xs:restriction>
      </xs:simpleType>
      <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>
      <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>
      <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>
      <xs:simpleType name="ServiceErrorType">
        <xs:restriction base="xs:string">
          <xs:enumeration value="UnknownError"/>
          <xs:enumeration value="ApplicationError"/>
          <xs:enumeration value="Timeout"/>
          <xs:enumeration value="ServiceBusy"/>
          <xs:enumeration value="SessionFull"/>
        </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="ClientActions">
        <xs:list>
          <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="Dismiss"/>
          </xs:restriction>
        </xs:list>
      </xs:simpleType>
    </xs:schema>
  </wsdl:types>
</wsdl:definitions>
```
<xs:enumeration value="Close"/>
<xs:enumeration value="OpenInClient"/>
<xs:enumeration value="Refresh"/>
</xs:restriction>
</xs:simpleType>
</xs:list>
</xs:complexType>
<xs:complexType name="AppServerInfo">
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="1" name="AppCapabilities" type="tns:BroadcastDictionaryType"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="PPTStateData">
<xs:sequence>
<xs:element minOccurs="1" maxOccurs="1" name="SlideId" type="xs:unsignedInt"/>
<xs:element minOccurs="0" maxOccurs="1" name="AtPosition" type="xs:double"/>
<xs:element minOccurs="0" maxOccurs="1" name="AnimationStepDataList" type="tns:ArrayOfPPTBroadcastAnimationStepData"/>
<xs:element minOccurs="0" maxOccurs="1" name="MediaStateDataList" type="tns:ArrayOfPPTBroadcastMediaStateData"/>
<xs:element minOccurs="1" maxOccurs="1" name="PPTSlideShowState" type="tns:PPTSlideShowState"/>
</xs:sequence>
</xs:complexType>
</xs:complexType>
<xs:complexType name="ArrayOfPPTBroadcastAnimationStepData">
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" name="PPTBroadcastAnimationStepData" nillable="true" type="tns:PPTBroadcastAnimationStepData"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="PPTBroadcastAnimationStepData">
<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>
<xs:complexType name="ArrayOfPPTBroadcastMediaStateData">
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" name="PPTBroadcastMediaStateData" nillable="true" type="tns:PPTBroadcastMediaStateData"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="PPTBroadcastMediaStateData">
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="1" name="MediaId" type="xs:string"/>
<xs:element minOccurs="1" maxOccurs="1" name="State" type="tns:PPTMediaPlaybackState"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="PPTMediaPlaybackState">
<xs:restriction base="xs:string">
<xs:enumeration value="Paused"/>
<xs:enumeration value="Playing"/>
<xs:enumeration value="Stopped"/>
</xs:restriction>
</xs:simpleType>
</xs:complexType>
<xs:complexType name="WordStateData">
<xs:sequence>
</xs:sequence>
</xs:complexType>
<soap12:operation
  style="document"/>
  <wSDL:input>
    <soap12:body use="literal"/>
  </wSDL:input>
  <wSDL:output>
    <soap12:body use="literal"/>
  </wSDL:output>
</wSDL:operation>
</wSDL:binding>
<wSDL:message name="BroadcastEndSessionSoapIn">
  <wSDL:part name="parameters" element="tns:BroadcastEndSession"/>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSDL:message>
</wSL
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 updates to those products.

- Microsoft Lync Server 2013
- Microsoft Skype for Business Server 2015
- Microsoft Skype for Business Server 2019

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

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

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

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

The revision class None means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

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

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Revision class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1</td>
<td>Normative References Updated [SOAP1.2-1/2007] and [SOAP1.2-2/2007] references throughout the document.</td>
<td>Minor</td>
</tr>
</tbody>
</table>
9 Index

A
Abstract data model
  server 22
Applicability 9
AppServerInfo complex type 11
ArrayOfPPTBroadcastAnimationStepData complex type 12
ArrayOfPPTBroadcastMediaStateData complex type 12
Attribute groups 20
Attributes 20

B
BroadcastAppCapability simple type 16
BroadcastDataKey simple type 17
BroadcastState simple type 18
BroadcastUser complex type 12

C
Capability negotiation 9
Change tracking 50
ClientActions simple type 18
Complex types 11
  AppServerInfo 11
  ArrayOfPPTBroadcastAnimationStepData 12
  ArrayOfPPTBroadcastMediaStateData 12
  BroadcastUser 12
  EnumTypes 13
  PPTBroadcastAnimationStepData 13
  PPTBroadcastMediaStateData 13
  PPTStateData 14
  ServiceError 14
  ServiceResult 15
  WordStateData 15

D
Data model - abstract
  server 22

E
EnumTypes complex type 13
Events
  local - server 36
  timer - server 35
Examples
  overview 37
  Presenter client 37

F
Fields - vendor-extensible 9
Full WSDL 42

G
Glossary 6

I
Implementer - security considerations 41
Index of security parameters 41
Informative references 8
Initialization
  server 23
Introduction 6

L
Local events
  server 36

M
Message processing
  server 23
Messages
  AppServerInfo complex type 11
  ArrayOfPPTBroadcastAnimationStepData complex type 12
  ArrayOfPPTBroadcastMediaStateData complex type 12
  attribute groups 20
  attributes 20
  BroadcastAppCapability simple type 16
  BroadcastDataKey simple type 17
  BroadcastState simple type 18
  BroadcastUser complex type 12
  ClientActions simple type 18
  complex types 11
  elements 10
  enumerated 10
  EnumTypes complex type 13
groups 20
  namespaces 10
  PPTBroadcastAnimationStepData complex type 13
  PPTBroadcastMediaStateData complex type 13
  PPTMediaPlaybackState simple type 19
  PPTSlideShowState simple type 19
  PPTStateData complex type 14
  ServiceError complex type 14
  ServiceErrorType simple type 20
  ServiceResult complex type 15
  simple types 16
  syntax 10
  transport 10
  WordStateData complex type 15

N
Namespaces 10
Normative references 7

O
Operations
  BroadcastEndSession 23
Overview (synopsis) 8

P

Parameters - security index 41
PPTBroadcastAnimationStepData complex type 13
PPTBroadcastMediaStateData complex type 13
PPTMediaPlaybackState simple type 19
PPTSlideShowState simple type 19
PPTStateData complex type 14
Preconditions 9
Prerequisites 9
Presenter client example 37
Product behavior 49
Protocol Details
   overview 21

R

References 7
   informative 8
   normative 7
Relationship to other protocols 8

S

Security
   implemtener considerations 41
   parameter index 41
Sequencing rules
   server 23
Server
   abstract data model 22
   BroadcastEndSession operation 23
   BroadcastGetAppCapabilities operation 25
   BroadcastPing operation 34
   BroadcastPutData operation 28
   BroadcastStartSession operation 31
   initialization 23
   local events 36
   message processing 23
   sequencing rules 23
   timer events 35
   timers 23
ServiceError complex type 14
ServiceErrorType simple type 20
ServiceResult complex type 15
Simple types 16
   BroadcastAppCapability 16
   BroadcastDataKey 17
   BroadcastState 18
   ClientActions 18
   PPTMediaPlaybackState 19
   PPTSlideShowState 19
   ServiceErrorType 20
   Standards assignments 9
Syntax
   messages - overview 10

T

Timer events
   server 35
   timers 23
Tracking changes 50
Transport 10
Types
   complex 11
   simple 16
Vendor-extensible fields 9
Versioning 9
WordStateData complex type 15
WSDL 42