[MS-RGSWS]:

Response Group Service Web Service Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
3/31/2010	0.1	Major	Initial Availability
4/30/2010	0.2	Editorial	Revised and edited the technical content
6/7/2010	0.3	Editorial	Revised and edited the technical content
6/29/2010	0.4	Editorial	Changed language and formatting in the technical content.
7/23/2010	0.4	No Change	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	1.0	Major	Significantly changed the technical content.
11/15/2010	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.0	Major	Significantly changed the technical content.
4/11/2012	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.1	Minor	Clarified the meaning of the technical content.
2/11/2013	2.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.1	No Change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.1	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.1	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.2	Minor	Clarified the meaning of the technical content.
7/31/2014	2.2	No Change	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.2	No Change	No changes to the meaning, language, or formatting of the technical content.
3/30/2015	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Response Group Web Service protocol specifies the procedure to access agent information exposed by a protocol server. This protocol enables third parties to build clients for the purpose of sign in or sign out of an agent and find information about which response group an agent belongs to.

The service exposes several methods to enable agents to interact with the system. For example, one method enables the agent to sign in and out of agent groups, enabling control if the service can send the agent calls.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

agent: A device that is connected to a computer network. Also referred to as an endpoint.

authentication: The act of proving an identity to a server while providing key material that binds the identity to subsequent communications.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms specified in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

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 over Secure Sockets Layer (HTTPS): An extension of **HTTP** that securely encrypts and decrypts webpage requests.

in-band provisioning: A process in which a protocol client obtains configuration information from a protocol server.

response group: An object that is used to route and queue incoming calls to a collection of agents who were designated to handle calls from a Response Group Service.

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data—a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client **authentication** using X.509 certificates (2). For more information, see [X509]. The SSL protocol is precursor to Transport Layer Security (TLS). The TLS version 1.0 specification is based on SSL version 3.0.

security identifier (SID): An identifier for security principals in Windows that is used to identify an account or a group. Conceptually, the **SID** is composed of an account authority portion (typically a domain) and a smaller integer representing an identity relative to the account authority, termed the relative identifier (RID). The **SID** format is specified in [MS-DTYP] section 2.4.2; a string representation of **SIDs** is specified in [MS-DTYP] section 2.4.2 and [MS-AZOD] section 1.1.1.2.

Session Initiation Protocol (SIP) address: A URI that does not include a "sip:" prefix and is used to establish multimedia communications sessions between two or more users over an IP network, as described in [RFC3261].

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 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 envelope: A container for **SOAP message** information and the root element of a **SOAP** document. See [SOAP1.2-1/2007] section 5.1 for more information.

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

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

web service: A unit of application logic that provides data and services to other applications and can be called by using standard Internet transport protocols such as **HTTP**, Simple Mail Transfer Protocol (SMTP), or File Transfer Protocol (FTP). Web services can perform functions that range from simple requests to complicated business processes.

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.

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

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-OCAUTHWS] Microsoft Corporation, "OC Authentication Web Service Protocol".

[MS-SIPREGE] Microsoft Corporation, "Session Initiation Protocol (SIP) Registration Extensions".

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

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

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

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

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

1.2.2 Informative References

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

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

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

1.3 Overview

This protocol enables a protocol client to access **agent** information exposed by a protocol server. The protocol client issues requests to a protocol server. The protocol server receives, processes, and responds to the requests of protocol clients.

This protocol provides the following functionality:

- Indicates if a user is an agent.
- Retrieves agent information.

  The protocol client can get information about the current user from the protocol server through the **Web service** described in this protocol.
- Retrieves the list of agent groups.

  The protocol client can retrieve the list of agent groups the currently logged-in user is part of. The object obtained contains information about the group.
- Signs in and signs out of agent groups.

The protocol client can send commands to the Web service described in this protocol to sign in and sign out the user from agent groups.

- Retrieves the list of active **response groups**.

  The protocol client can retrieve the list of response groups the user is part of through agent groups. The object obtained contains information about the response groups.

The Web service methods are documented in detail in section 6.

1.4 Relationship to Other Protocols

This protocol uses the **Simple Object Access Protocol (SOAP)** message protocol for formatting requests and responses, as described in [SOAP1.2/1] and [SOAP1.2/2]. It transmits messages using the **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [RFC2818].

The following figure shows the underlying messaging and transport stack that the protocol uses.

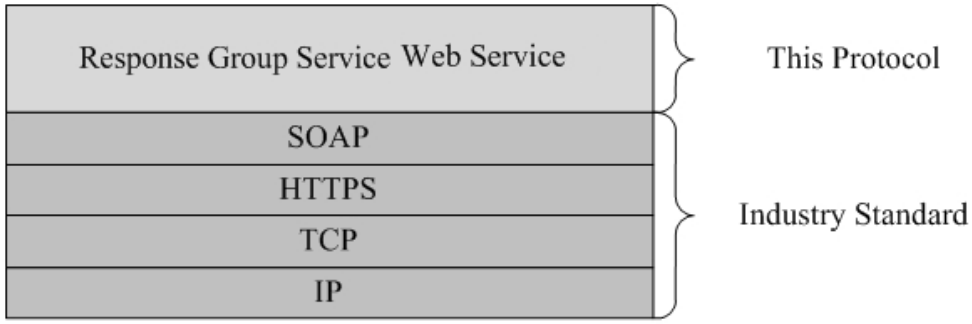

Figure 1: Underlying message and transport stack

1.5 Prerequisites/Preconditions

The protocol client can obtain the **Uniform Resource Locator (URL)** of the protocol server by using the mechanism described in [MS-SIPREGE] section 2.2.2.

This protocol requires that the protocol client has the correct permissions to call the methods on the protocol server.

The protocol client and server are described in [MS-OCAUTHWS].

This protocol requires **authentication** as described in [MS-OCAUTHWS].

1.6 Applicability Statement

This protocol is used to retrieve agent information.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers MUST support SOAP over HTTPS. The client can obtain the address of the protocol server via an **in-band provisioning** response, as specified in [MS-SIPREGE] section 2.2.2.5.1.

2.2 Common Message Syntax

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

2.2.1 Namespaces

This protocol specifies and references **XML namespace** using the mechanisms specified in [XMLNS]. Although this document associates an **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.

The following table specifies which XML namespace prefix is associated with each XML namespace that is used.

Prefix	Namespace URI	Reference
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
tns	http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy	
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
s1	http://microsoft.com/wsdl/types/	

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.

Complex type	Description
AcdAgent	Information about the authenticated user.

Complex type	Description
ArrayOfGuid	A list of **GUID** instances.

2.2.4.1 AcdAgent

The **AcdAgent** complex type contains information related to an authenticated user.

```
<s:complexType name="AcdAgent">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="Id" type="s1:guid" />
    <s:element minOccurs="0" maxOccurs="1" name="UserSid" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="SipAddress" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="DisplayName" type="s:string" />
  </s:sequence>
</s:complexType>
```

Id: Unique identifier identifying the user.

UserSid: The **security identifier (SID)** of the authenticated user.

SipAddress: Is built from the **Session Initiation Protocol (SIP) address** of the authenticated user plus the "sip:" prefix.

DisplayName: Display name of the authenticated user.

Elements having *minOccurs="0"* are not present in case the server fails to retrieve the information related to the authenticated user.

2.2.4.2 ArrayOfGuid

The **ArrayOfGuid** complex type is a list of GUIDs representing groups.

```
<s:complexType name="ArrayOfGuid">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="guid" type="s1:guid" />
  </s:sequence>
</s:complexType>
```

guid: Identifier of a group. It is not present in case the server fails to retrieve the information related to the authenticated user.

2.2.5 Simple Types

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

Simple type	Description
guid	The **guid** simple type represents a GUID.

2.2.5.1 guid

Namespace: http://microsoft.com/wsdl/types/

A GUID that is defined as follows:

```
<s:simpleType name="guid">
  <s:restriction base="s:string">
    <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-
F]{12}"/>
  </s:restriction>
</s:simpleType>
```

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.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

3 Protocol Details

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

There is no order in which methods are required to be called, except that at least one of the sign in methods, **SignIn** (section 3.1.4.5) and **SignInMultiple** (section 3.1.4.6), MUST precede the sign out methods, **SignOut** (section 3.1.4.7) and **SignOutMultiple** (section 3.1.4.8). The sign in and sign out methods require as input valid GUIDs that can be retrieved using the **GetGroups** method (section 3.1.4.3). If the user is not an agent, all the methods from the protocol having a Boolean as a return value will return false.

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC2616] (Section 10, Status Code Definitions). This protocol allows protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults using HTTP status codes.

3.1 RgsAgentService Server Details

This protocol does not require the protocol server to keep any states and information about the protocol client, unless required by the authentication mechanism defined in [MS-OCAUTHWS] section 3.

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

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

Operation	Description
GetActiveResponseGroups	Retrieve the list of active response groups the authenticated user is part of.
GetAgent	Retrieve the information about the authenticated agent.
GetGroups	Retrieve the groups the authenticated user is part of.
IsAgent	Indicate if the authenticated user is an agent or not.
SignIn	Sign in the authenticated user to the specified group.
SignInMultiple	Sign in the authenticated user to the specified groups.

Operation	Description
SignOut	Sign out the authenticated user from the specified group.
SignOutMultiple	Sign out the authenticated user from the specified groups.

3.1.4.1 GetActiveResponseGroups

The following excerpt from this protocol's WSDL specifies the messages that constitute the **GetActiveResponseGroups** operation.<1>

```
<wsdl:operation name="GetActiveResponseGroups">
  <wsdl:input message="tns:GetActiveResponseGroupsSoapIn" />
  <wsdl:output message="tns:GetActiveResponseGroupsSoapOut" />
</wsdl:operation>
```

3.1.4.1.1 Messages

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

Message	Description
GetActiveResponseGroupsSoapIn	Request from a client to retrieve the list of response groups the user is part of.
GetActiveResponseGroupsSoapOut	Response to a request to retrieve the list of response groups the user is part of.

3.1.4.1.1.1 GetActiveResponseGroupsSoapIn

The **GetActiveResponseGroupsSoapIn** **SOAP message** is a request that is sent from the protocol client to retrieve the list of response groups the user is part of. The request information MUST be captured in the **GetActiveResponseGroups** element in the **SOAP body** of the message. The **GetActiveResponseGroups** element is specified in section 3.1.4.1.2.1.

3.1.4.1.1.2 GetActiveResponseGroupsSoapOut

The **GetActiveResponseGroupsSoapOut** SOAP message is a response that is sent by the protocol server. This message contains information about the response groups the user is part of. The information is included in the complex type **ArrayOfResponseGroupEntry**, as specified in section 3.1.4.1.3.1. If the user is not an agent, the response SHOULD NOT contain any complex type.

3.1.4.1.2 Elements

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

Element	Description
GetActiveResponseGroups	The overall container of the request to retrieve response groups.
GetActiveResponseGroupsResponse	The overall container of the response to the request to

Element	Description
	retrieve response groups.

3.1.4.1.2.1 GetActiveResponseGroups

The **GetActiveResponseGroups** element is the overall container of the information that is sent in the SOAP request to retrieve the response groups the user is part of. A protocol client MUST adhere to the following schema.

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

3.1.4.1.2.2 GetActiveResponseGroupsResponse

The **GetActiveResponseGroupsResponse** element is the overall container in the response to the **GetActiveResponseGroups** (section 3.1.4.1) request. **GetActiveResponseGroups** encapsulates the information about the authenticated user. A protocol server MUST adhere to the following schema for this element within the **SOAP envelope**.

```
  <s:element name="GetActiveResponseGroupsResponse">
    <s:complexType>
   <s:sequence>
     <s:element minOccurs="0" maxOccurs="1" name="GetActiveResponseGroupsResult"
type="tns:ArrayOfResponseGroupEntry" />
   </s:sequence>
    </s:complexType>
  </s:element>
```

GetActiveResponseGroupsResult: An array of **ResponseGroupEntry** complex types (section 3.1.4.1.3.2).

3.1.4.1.3 Complex Types

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

Complex type	Description
ArrayOfResponseGroupEntry	A list of **ResponseGroupEntry** type objects.
ResponseGroupEntry	Information about a response group.

3.1.4.1.3.1 ArrayOfResponseGroupEntry

The **ArrayOfResponseGroupEntry** complex type is a list of **ResponseGroupEntry** complex types (section 3.1.4.1.3.2) that the current authenticated user is part of.

```
<s:complexType name="ArrayOfResponseGroupEntry">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="ResponseGroupEntry" nillable="true"
type="tns:ResponseGroupEntry"/>
  </s:sequence>
```

```
</s:complexType>
```

ResponseGroupEntry: See section 3.1.4.1.3.2.

3.1.4.1.3.2 ResponseGroupEntry

The **ResponseGroupEntry** complex type contains information related to a response group.

```
<s:complexType name="ResponseGroupEntry">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="Uri" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="DisplayName" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="IsAnonymized" type="s:boolean" />
    <s:element minOccurs="1" maxOccurs="1" name="IsOutboundAllowed" type="s:boolean" />
  </s:sequence>
</s:complexType>
```

Uri: Represents the Session Initiation Protocol (SIP) address of the response group plus the "sip:" prefix.

DisplayName: Display name of the response group.

IsAnonymized: **Boolean** indicating if the response group supports anonymization.

IsOutboundAllowed: **Boolean** indicating if the response group allows its agent to make outbound calls.

Elements having *minOccurs="0"* are not present in case the server fails to retrieve the information related to the authenticated user.

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 GetAgent

The following excerpt from this protocol's WSDL specifies the messages that constitute the **GetAgent** operation.

```
<wsdl:operation name="GetAgent">
      <wsdl:input message="tns:GetAgentSoapIn" />
      <wsdl:output message="tns:GetAgentSoapOut" />
</wsdl:operation>
```

3.1.4.2.1 Messages

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

Message	Description
GetAgentSoapIn	Request from a client to retrieve information about the authenticated user.
GetAgentSoapOut	Response to a request to retrieve information about the authenticated user.

3.1.4.2.1.1 GetAgentSoapIn

The **GetAgentSoapIn** SOAP message is a request that is sent from the protocol client to retrieve information about the authenticated user. The request information MUST be captured in the **GetAgent** element in the SOAP body of the message. The **GetAgent** element is specified in section 3.1.4.2.2.1.

3.1.4.2.1.2 GetAgentSoapOut

The **GetAgentSoapOut** SOAP message is a response that is sent by the protocol server. This message contains information about the authenticated user if the user is an agent. The information is included in the complex type **AcdAgent**, which is specified in section 2.2.4.1. If the user is not an agent, the response SHOULD NOT contain any complex type.

3.1.4.2.2 Elements

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

Element	Description
GetAgent	The overall container of the request to retrieve agent information.
GetAgentResponse	The overall container of the response to the request to retrieve agent information.

3.1.4.2.2.1 GetAgent

The **GetAgent** element is the overall container of the information that is sent in the SOAP request to retrieve agent information. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

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

3.1.4.2.2.2 GetAgentResponse

The **GetAgentResponse** complex type is the overall container in the response to the **GetAgent** (section 3.1.4.2) request. **GetAgentResponse** encapsulates the information about the authenticated user. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="GetAgentResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="GetAgentResult" type="tns:AcdAgent" />
    </s:sequence>
  </s:complexType>
</s:element>
```

GetAgentResult: An **AcdAgent** object which describes the authenticated agent. **AcdAgent** is described in section 2.2.4.1.

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 GetGroups

The following excerpt from this protocol's WSDL specifies the messages that constitute the **GetGroups** operation.

```
<wsdl:operation name="GetGroups">
      <wsdl:input message="tns:GetGroupsSoapIn" />
      <wsdl:output message="tns:GetGroupsSoapOut" />
</wsdl:operation>
```

3.1.4.3.1 Messages

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

Message	Description
GetGroupsSoapIn	Request from a client to retrieve the groups of an authenticated user.
GetGroupsSoapOut	Response to a request to retrieve the groups of an authenticated user.

3.1.4.3.1.1 GetGroupsSoapIn

The **GetGroupsSoapIn** SOAP message is a request that is sent from the protocol client to retrieve the groups of an authenticated user. The request information MUST be captured in the **GetGroups** element in the SOAP body of the message. The **GetGroups** element is specified in section 3.1.4.3.2.1.

3.1.4.3.1.2 GetGroupsSoapOut

The **GetGroupsSoapOut** SOAP message is a response that is sent by the protocol server. This message contains a list of groups that the agent is member of. The information is included in the complex type **ArrayOfAcdGroup**, which is specified in section 3.1.4.3.3.3. If the user is not an agent, the **ArrayOfAcdGroup** complex type SHOULD be empty.

3.1.4.3.2 Elements

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

Element	Description
GetGroups	The overall container of the request to retrieve agent groups.
GetGroupsResponse	The overall container of the response to the request to retrieve agent groups.

3.1.4.3.2.1 GetGroups

The **GetGroups** element is the overall container of the information that is sent in the SOAP request to retrieve agent groups. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

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

3.1.4.3.2.2 GetGroupsResponse

The **GetGroupsResponse** complex type is the overall container in the response to the **GetGroups** (section 3.1.4.3) request. **GetGroupsResponse** encapsulates the information about the groups an agent is part of. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="GetGroupsResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="GetGroupsResult" type="tns:ArrayOfAcdGroup"
/>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetGroupsResult: An array of **AcdGroup** objects which represent the groups where the authenticated agent is a member. **AcdGroup** is described in section 3.1.4.3.3.1.

3.1.4.3.3 Complex Types

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

Complex type	Description
AcdGroup	Information about a group.
ArrayOfAcdAgent	A list of agents belonging to a group.
ArrayOfAcdGroup	A list of **AcdGroups**.

3.1.4.3.3.1 AcdGroup

The **AcdGroup** complex type contains information related to an agent group.

```
<s:complexType name="AcdGroup">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="Id" type="s1:guid" />
    <s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="NumberOfAgents" type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="CanSignIn" type="s:boolean" />
    <s:element minOccurs="1" maxOccurs="1" name="SignInState" type="tns:SignInState" />
    <s:element minOccurs="0" maxOccurs="1" name="AllAgents"     type="tns:ArrayOfAcdAgent" />
    <s:element minOccurs="1" maxOccurs="1" name="NumberOfCallsWaiting" type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="LongestWaitingTime" type="s:int" />

  </s:sequence>
</s:complexType>
```

Id: Unique identifier identifying the group.

Name: Name of the group.

NumberOfAgents: Number of agents who are members of the group. This element SHOULD be greater than or equal to zero.

CanSignIn: A **Boolean** that indicates if the group supports sign in and sign out.

SignInState: State of the current user for this group.

AllAgents: List of agents who are members of the group.

NumberOfCallsWaiting: Number of calls waiting in the queues served by the group. This element SHOULD be greater than or equal to zero.

LongestWaitingTime: The longest waiting time in seconds among all the calls waiting in the queues served by the group. This element SHOULD be greater than or equal to zero.

Elements having *minOccurs="0"* are not present in case the server fails to retrieve the information related to the authenticated user.

3.1.4.3.3.2 ArrayOfAcdAgent

The **ArrayOfAcdAgent** complex type is a list of agents who are members of a group.

```
<s:complexType name="ArrayOfAcdAgent">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="AcdAgent" nillable="true"
type="tns:AcdAgent" />
```

```
  </s:sequence>
</s:complexType>
```

AcdAgent: A complex type containing information related to an authenticated user, as specified in section 2.2.4.1.

3.1.4.3.3.3 ArrayOfAcdGroup

The **ArrayOfAcdGroup** complex type is a list of **AcdGroups** (section 3.1.4.3.3.1) the current authenticated user is part of.

```
<s:complexType name="ArrayOfAcdGroup">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="AcdGroup" nillable="true"
type="tns:AcdGroup"/>
  </s:sequence>
</s:complexType>
```

AcdGroup: A complex type containing information related to an agent group, as specified in section 3.1.4.3.3.1.

3.1.4.3.4 Simple Types

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

Simple type	Description
SignInState	The sign-in state of an agent in a given group.

3.1.4.3.4.1 SignInState

The **SignInState** simple type represents the state of a user in a given group.

```
<s:simpleType name="SignInState">
  <s:restriction base="s:string">
    <s:enumeration value="SignedIn" />
    <s:enumeration value="SignedOut" />
    <s:enumeration value="Unknown" />
  </s:restriction>
</s:simpleType>
```

SignedIn: The user is currently signed in to the group.

SignedOut: The user is currently signed out of the group.

Unknown: The user's state cannot be retrieved.

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 IsAgent

The following excerpt from this protocol's WSDL specifies the messages that constitute the **IsAgent** operation:

```
<wsdl:operation name="IsAgent">
      <wsdl:input message="tns:IsAgentSoapIn" />
      <wsdl:output message="tns:IsAgentSoapOut" />
</wsdl:operation>
```

3.1.4.4.1 Messages

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

Message	Description
IsAgentSoapIn	Request from a client to retrieve information about whether the user is an agent.
IsAgentSoapOut	Response to a request to retrieve information about whether the user is an agent.

3.1.4.4.1.1 IsAgentSoapIn

The **IsAgentSoapIn** SOAP message is a request that is sent from the protocol client to retrieve information about whether the user is an agent. The request information MUST be captured in the **IsAgent** element in the SOAP body of the message. The **IsAgent** element is specified in section 3.1.4.4.2.1.

3.1.4.4.1.2 IsAgentSoapOut

The **IsAgentSoapOut** SOAP message is a response that is sent by the protocol server. This message MUST contain a **Boolean** that indicates if the user is an agent or not.

3.1.4.4.2 Elements

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

Element	Description
IsAgent	The overall container of the request to retrieve agent state.
IsAgentResponse	The overall container of the response to the request to retrieve agent state.

3.1.4.4.2.1 IsAgent

The **IsAgent** element is the overall container of the information that is sent in the SOAP request to retrieve agent state. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

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

3.1.4.4.2.2 IsAgentResponse

The **IsAgentResponse** element is the overall container in the response to the **IsAgent** (section 3.1.4.4) request. **IsAgentResponse** encapsulates the information about the agent state. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="IsAgentResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="IsAgentResult" type="s:boolean" />
    </s:sequence>
  </s:complexType>
</s:element>
```

IsAgentResult: Indicates whether the authenticated user is a response group agent.

3.1.4.4.3 Complex Types

None.

3.1.4.4.4 Simple Types

None.

3.1.4.4.5 Attributes

None.

3.1.4.4.6 Groups

None.

3.1.4.4.7 Attribute Groups

None.

3.1.4.5 SignIn

The following excerpt from this protocol's WSDL specifies the messages that constitute the **SignIn** operation:

```
<wsdl:operation name="SignIn">
      <wsdl:input message="tns:SignInSoapIn" />
      <wsdl:output message="tns:SignInSoapOut" />
</wsdl:operation>
```

3.1.4.5.1 Messages

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

Message	Description
SignInSoapIn	A request from a client to sign in the user in the specified group.
SignInSoapOut	A response to a request to sign in the user in the specified group.

3.1.4.5.1.1 SignInSoapIn

The **SignInSoapIn** SOAP message is a request that is sent from the protocol client to sign in the user in the specified group. The request information MUST be captured in the **SignIn** element in the SOAP body of the message. The **SignIn** element is specified in section 3.1.4.5.2.1.

3.1.4.5.1.2 SignInSoapOut

This SOAP message is a response that is sent by the protocol server. This message MUST contain a **Boolean** indicating if the operation has succeeded.

3.1.4.5.2 Elements

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

Element	Description
SignIn	The overall container of the request to sign in the user to a group.
SignInResponse	The overall container of the response to the request to sign in the user to a group.

3.1.4.5.2.1 SignIn

The **SignIn** element is the overall container of the information that is sent in the SOAP request to sign in the user in a group. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignIn">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="groupId" type="s1:guid" />
    </s:sequence>
  </s:complexType>
</s:element>
```

groupId: The identifier of the group to sign in the authenticated user.

3.1.4.5.2.2 SignInResponse

The **SignInResponse** element is the overall container in the response to the **SignIn** (section 3.1.4.5) request. **SignInResponse** contains the result of the operation. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignInResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="SignInResult" type="s:boolean" />
    </s:sequence>
  </s:complexType>
</s:element>
```

SignInResult: Indicates whether the authenticated user successfully signed into the given group.

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.

3.1.4.6 SignInMultiple

The following excerpt from this protocol's WSDL specifies the messages that constitute the **SignInMultiple** operation:

```
<wsdl:operation name="SignInMultiple">
      <wsdl:input message="tns:SignInMultipleSoapIn" />
      <wsdl:output message="tns:SignInMultipleSoapOut" />
</wsdl:operation>
```

3.1.4.6.1 Messages

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

Message	Description
SignInMultipleSoapIn	A request from a client to sign in the user in the specified groups.
SignInMultipleSoapOut	A response to a request to sign in the user in the specified groups.

3.1.4.6.1.1 SignInMultipleSoapIn

The **SignInMultipleSoapIn** SOAP message is a request that is sent from the protocol client to sign in the user in the specified groups. The request information MUST be captured in the **SignInMultiple** element in the SOAP body of the message. The **SignInMultiple** element is specified in section 3.1.4.6.2.1.

3.1.4.6.1.2 SignInMultipleSoapOut

The **SignInMultipleSoapOut** SOAP message is a response that is sent by the protocol server. This message MUST contain a **Boolean** indicating if the operation succeeded.

3.1.4.6.2 Elements

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

Element	Description
SignInMultiple	The overall container of the request to sign in the user in multiple groups.
SignInMultipleResponse	The overall container of the response to the request to sign in the user in multiple groups.

3.1.4.6.2.1 SignInMultiple

The **SignInMultiple** element is the overall container of the information that is sent in the SOAP request to sign in the user in multiple groups. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignInMultiple">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="groupIds" type="tns:ArrayOfGuid" />
    </s:sequence>
  </s:complexType>
</s:element>
```

groupIds: An array of identifiers of the groups to sign the authenticated user into.

3.1.4.6.2.2 SignInMultipleResponse

The **SignInMultipleResponse** element is the overall container in the response to the **SignInMultiple** (section 3.1.4.6) request. **SignInMultipleResponse** contains the result of the operation. If the operation partially succeeded, the return value SHOULD be set to **FALSE**. Information about the current sign-in state can be retrieved using the **GetGroups** message, as defined in section 3.1.4.3. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignInMultipleResponse">
  <s:complexType>
    <s:sequence>
```

```
        <s:element minOccurs="1" maxOccurs="1" name="SignInMultipleResult" type="s:boolean" />
      </s:sequence>
    </s:complexType>
  </s:element>
```

SignInMultipleResult: Indicates whether the authenticated user successfully signed into all given groups.

3.1.4.6.3 Complex Types

None.

3.1.4.6.4 Simple Types

None.

3.1.4.6.5 Groups

None.

3.1.4.6.6 Attributes

None.

3.1.4.6.7 Attribute Groups

None.

3.1.4.7 SignOut

The following excerpt from this protocol's WSDL specifies the messages that constitute the **SignOut** operation:

```
<wsdl:operation name="SignOut">
      <wsdl:input message="tns:SignOutSoapIn" />
      <wsdl:output message="tns:SignOutSoapOut" />
</wsdl:operation>
```

3.1.4.7.1 Messages

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

Message	Description
SignOutSoapIn	Request from a client to sign out the user from the specified group.
SignOutSoapOut	Response to a request to sign out the user from the specified group.

3.1.4.7.1.1 SignOutSoapIn

The **SignOutSoapIn** SOAP message is a request that is sent from the protocol client to sign out the user from the specified group. The request information MUST be captured in the **SignOut** element in the SOAP body of the message. The **SignOut** element is specified in section 3.1.4.7.2.1.

3.1.4.7.1.2 SignOutSoapOut

The **SignOutSoapOut** SOAP message is a response that is sent by the protocol server. This message MUST contain a **Boolean** indicating if the operation succeeded.

3.1.4.7.2 Elements

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

Element	Description
SignOut	The overall container of the request to sign out the user from a group.
SignOutResponse	The overall container of the response to the request to sign out the user from a group.

3.1.4.7.2.1 SignOut

The **SignOut** element is the overall container of the information that is sent in the SOAP request to sign out the user from a group. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignOut">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="groupId" type="s1:guid" />
    </s:sequence>
  </s:complexType>
</s:element>
```

groupId: Identifier of the group to sign the authenticated user out of.

3.1.4.7.2.2 SignOutResponse

The **SignOutResponse** element is the overall container in the response to the **SignOut** (section 3.1.4.7.2.1) request. **SignOutResponse** contains the result of the operation. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignOutResponse">
  <s:complexType>
     <s:sequence>
       <s:element minOccurs="1" maxOccurs="1" name="SignOutResult" type="s:boolean" />
     </s:sequence>
  </s:complexType>
</s:element>
```

SignOutResult: Indicates whether the authenticated user successfully signed out from the given group.

3.1.4.7.3 Complex Types

None.

3.1.4.7.4 Simple Types

None.

3.1.4.7.5 Attributes

None.

3.1.4.7.6 Groups

None.

3.1.4.7.7 Attribute Groups

None.

3.1.4.8 SignOutMultiple

The following excerpt from this protocol's WSDL specifies the messages that constitute the **SignOutMultiple** operation.

```
<wsdl:operation name="SignOutMultiple">
      <wsdl:input message="tns:SignOutMultipleSoapIn" />
      <wsdl:output message="tns:SignOutMultipleSoapOut" />
</wsdl:operation>
```

3.1.4.8.1 Messages

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

Message	Description
SignOutMultipleSoapIn	Request from a client to sign out the user from the specified groups.
SignOutMultipleSoapOut	Response to a request to sign out the user from the specified groups.

3.1.4.8.1.1 SignOutMultipleSoapIn

The **SignOutMultipleSoapIn** SOAP message is a request that is sent from the protocol client to sign out the user from the specified groups. The request information MUST be captured in the **SignOutMultiple** element in the SOAP body of the message. The **SignOutMultiple** element is specified in section 3.1.4.8.2.1.

3.1.4.8.1.2 SignOutMultipleSoapOut

The **SignOutMultipleSoapOut** SOAP message is a response that is sent by the protocol server. This message MUST contain a **Boolean** indicating if the operation succeeded.

3.1.4.8.2 Elements

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

Element	Description
SignOutMultiple	The overall container of the request to sign out the user from multiple groups.
SignOutMultipleResponse	The overall container of the response to the request to sign out the user from multiple groups.

3.1.4.8.2.1 SignOutMultiple

The **SignOutMultiple** element is the overall container of the information that is sent in the SOAP request to sign out the user from multiple groups. A protocol client MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignOutMultiple">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="groupIds" type="tns:ArrayOfGuid" />
    </s:sequence>
  </s:complexType>
</s:element>
```

groupIds: An array of identifiers of the groups in which we request to sign out the authenticated user.

3.1.4.8.2.2 SignOutMultipleResponse

The **SignOutMultipleResponse** element is the overall container in the response to the **SignOutMultiple** (section 3.1.4.8) request. **SignOutMultipleResponse** contains the result of the operation. If the operation partially succeeded, the return value SHOULD be set to **FALSE**. Information about the current sign-in state can be retrieved using the **GetGroups** message, as defined in section 3.1.4.3. A protocol server MUST adhere to the following schema for this element within the SOAP envelope.

```
<s:element name="SignOutMultipleResponse">
  <s:complexType>
     <s:sequence>
       <s:element minOccurs="1" maxOccurs="1" name="SignOutMultipleResult" type="s:boolean" />
     </s:sequence>
  </s:complexType>
</s:element>
```

SignOutMultipleResult: A boolean value which indicates if the authenticated user successfully sign out from the all given groups.

3.1.4.8.3 Complex Types

None.

3.1.4.8.4 Simple Types

None.

3.1.4.8.5 Attributes

None.

3.1.4.8.6 Groups

None.

3.1.4.8.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

4.1 Successful GetActiveResponseGroups Request and Response

The following example is a **GetActiveResponseGroups** request. This request is sent from a client to the server as a SOAP HTTPS request.

```
<soap:Body>
  <GetActiveResponseGroups
xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy"/>
</soap:Body>
```

This request results in the following successful SOAP HTTPS response.

```
<soap:Body>
  <GetActiveResponseGroupsResponse
xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <GetActiveResponseGroupsResult>
      <ResponseGroupEntry>
        <Uri>sip:helpdesk@contoso.com</Uri>
        <DisplayName>Contoso Helpdesk</DisplayName>
        <IsAnonymized>true</IsAnonymized>
        <IsOutboundAllowed>true</IsOutboundAllowed>
      </ResponseGroupEntry>
      <ResponseGroupEntry>
        <Uri>sip:hr@contoso.com</Uri>
        <DisplayName>Contoso HR</DisplayName>
        <IsAnonymized>true</IsAnonymized>
        <IsOutboundAllowed>false</IsOutboundAllowed>
      </ResponseGroupEntry>
    </GetActiveResponseGroupsResult>
  </GetActiveResponseGroupsResponse>
</soap:Body>
```

4.2 Successful SignIn Request and Response

The following example is a **SignIn** request. This request is sent from a client to the server as a SOAP HTTPS request.

```
<soap:Body>
  <SignIn xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <groupId>0762e30f-a76c-4c75-9390-4c07a9f4e51f</groupId>
 </SignIn>
</soap:Body>
```

This request results in the following successful SOAP HTTPS response.

```
<soap:Body>
  <SignInResponse
xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <SignInResult>true</SignInResult>
  </SignInResponse>
</soap:Body>
```

4.3 Unsuccessful SignOut Request and Response

The following example is a **SignOut** request. This request is sent from a client to the server as a SOAP HTTPS request.

```
<soap:Body>
  <SignOut xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <groupId>0762e30f-a76c-4c75-9390-4c07a9f4e51f</groupId>
  </SignOut>
</soap:Body>
```

This request results in the following SOAP HTTPS response.

```
<soap:Body>
  <SignOutResponse
xmlns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <SignOutResult>false</SignOutResult>
  </SignOutResponse>
</soap:Body>
```

5 Security

5.1 Security Considerations for Implementers

This protocol allows **Hypertext Transfer Protocol (HTTP)** connections only over **Secure Sockets Layer (SSL)**. Users are authenticated using the mechanism described in [MS-OCAUTHWS].

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL and schema are provided in this appendix.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy"
xmlns:s1="http://microsoft.com/wsdl/types/" xmlns:s="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
<wsdl:types>
  <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy">
    <s:import namespace="http://microsoft.com/wsdl/types/" />
    <s:element name="IsAgent">
      <s:complexType />
    </s:element>
    <s:element name="IsAgentResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="IsAgentResult" type="s:boolean" />
        </s:sequence>
       </s:complexType>
      </s:element>
    <s:element name="GetAgent">
      <s:complexType />
    </s:element>
    <s:element name="GetAgentResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="GetAgentResult" type="tns:AcdAgent" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:complexType name="AcdAgent">
      <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="Id" type="s1:guid" />
        <s:element minOccurs="0" maxOccurs="1" name="UserSid" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="SipAddress" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="DisplayName" type="s:string" />
      </s:sequence>
    </s:complexType>
    <s:element name="GetGroups">
      <s:complexType />
    </s:element>
    <s:element name="GetGroupsResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="GetGroupsResult"
type="tns:ArrayOfAcdGroup" />
         </s:sequence>
      </s:complexType>
    </s:element>
    <s:complexType name="ArrayOfAcdGroup">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="AcdGroup" nillable="true"
type="tns:AcdGroup" />
      </s:sequence>
    </s:complexType>
    <s:complexType name="AcdGroup">
      <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="Id" type="s1:guid" />
        <s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="NumberOfAgents" type="s:int" />
        <s:element minOccurs="1" maxOccurs="1" name="CanSignIn" type="s:boolean" />
        <s:element minOccurs="1" maxOccurs="1" name="SignInState" type="tns:SignInState" />
        <s:element minOccurs="0" maxOccurs="1" name="AllAgents" type="tns:ArrayOfAcdAgent" />
```

```
        <s:element minOccurs="1" maxOccurs="1" name="NumberOfCallsWaiting" type="s:int" />
        <s:element minOccurs="1" maxOccurs="1" name="LongestWaitingTime" type="s:int" />

      </s:sequence>
    </s:complexType>
    <s:simpleType name="SignInState">
      <s:restriction base="s:string">
        <s:enumeration value="SignedIn" />
        <s:enumeration value="SignedOut" />
        <s:enumeration value="Unknown" />
      </s:restriction>
    </s:simpleType>
    <s:complexType name="ArrayOfAcdAgent">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="AcdAgent" nillable="true"
type="tns:AcdAgent" />
      </s:sequence>
    </s:complexType>
    <s:element name="SignIn">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="groupId" type="s1:guid" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignInResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SignInResult" type="s:boolean" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignOut">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="groupId" type="s1:guid" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignOutResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SignOutResult" type="s:boolean" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignInMultiple">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="groupIds" type="tns:ArrayOfGuid" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:complexType name="ArrayOfGuid">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="guid" type="s1:guid" />
      </s:sequence>
    </s:complexType>
    <s:element name="SignInMultipleResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SignInMultipleResult" type="s:boolean"
/>
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignOutMultiple">
      <s:complexType>
        <s:sequence>
```

```
          <s:element minOccurs="0" maxOccurs="1" name="groupIds" type="tns:ArrayOfGuid" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SignOutMultipleResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SignOutMultipleResult"
type="s:boolean" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="GetActiveResponseGroups">
      <s:complexType />
    </s:element>
    <s:element name="GetActiveResponseGroupsResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="GetActiveResponseGroupsResult"
type="tns:ArrayOfResponseGroupEntry" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:complexType name="ArrayOfResponseGroupEntry">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="ResponseGroupEntry"
nillable="true" type="tns:ResponseGroupEntry" />
      </s:sequence>
    </s:complexType>
    <s:complexType name="ResponseGroupEntry">
      <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="Uri" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="DisplayName" type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="IsAnonymized" type="s:boolean" />
        <s:element minOccurs="1" maxOccurs="1" name="IsOutboundAllowed" type="s:boolean" />
      </s:sequence>
    </s:complexType>
  </s:schema>
  <s:schema elementFormDefault="qualified"
targetNamespace="http://microsoft.com/wsdl/types/">
    <s:simpleType name="guid">
      <s:restriction base="s:string">
        <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-
fA-F]{12}" />
      </s:restriction>
    </s:simpleType>
  </s:schema>
</wsdl:types>
<wsdl:message name="IsAgentSoapIn">
  <wsdl:part name="parameters" element="tns:IsAgent" />
</wsdl:message>
<wsdl:message name="IsAgentSoapOut">
  <wsdl:part name="parameters" element="tns:IsAgentResponse" />
</wsdl:message>
<wsdl:message name="GetAgentSoapIn">
  <wsdl:part name="parameters" element="tns:GetAgent" />
</wsdl:message>
<wsdl:message name="GetAgentSoapOut">
  <wsdl:part name="parameters" element="tns:GetAgentResponse" />
</wsdl:message>
<wsdl:message name="GetGroupsSoapIn">
  <wsdl:part name="parameters" element="tns:GetGroups" />
</wsdl:message>
<wsdl:message name="GetGroupsSoapOut">
  <wsdl:part name="parameters" element="tns:GetGroupsResponse" />
</wsdl:message>
<wsdl:message name="SignInSoapIn">
  <wsdl:part name="parameters" element="tns:SignIn" />
</wsdl:message>
```

```
<wsdl:message name="SignInSoapOut">
  <wsdl:part name="parameters" element="tns:SignInResponse" />
</wsdl:message>
<wsdl:message name="SignOutSoapIn">
  <wsdl:part name="parameters" element="tns:SignOut" />
</wsdl:message>
<wsdl:message name="SignOutSoapOut">
  <wsdl:part name="parameters" element="tns:SignOutResponse" />
</wsdl:message>
<wsdl:message name="SignInMultipleSoapIn">
  <wsdl:part name="parameters" element="tns:SignInMultiple" />
</wsdl:message>
<wsdl:message name="SignInMultipleSoapOut">
  <wsdl:part name="parameters" element="tns:SignInMultipleResponse" />
</wsdl:message>
<wsdl:message name="SignOutMultipleSoapIn">
  <wsdl:part name="parameters" element="tns:SignOutMultiple" />
</wsdl:message>
<wsdl:message name="SignOutMultipleSoapOut">
  <wsdl:part name="parameters" element="tns:SignOutMultipleResponse" />
</wsdl:message>
<wsdl:message name="GetActiveResponseGroupsSoapIn">
  <wsdl:part name="parameters" element="tns:GetActiveResponseGroups" />
</wsdl:message>
<wsdl:message name="GetActiveResponseGroupsSoapOut">
  <wsdl:part name="parameters" element="tns:GetActiveResponseGroupsResponse" />
</wsdl:message>
<wsdl:portType name="ProxyServiceSoap">
  <wsdl:operation name="IsAgent">
    <wsdl:input message="tns:IsAgentSoapIn" />
    <wsdl:output message="tns:IsAgentSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetAgent">
    <wsdl:input message="tns:GetAgentSoapIn" />
    <wsdl:output message="tns:GetAgentSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetGroups">
    <wsdl:input message="tns:GetGroupsSoapIn" />
    <wsdl:output message="tns:GetGroupsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="SignIn">
    <wsdl:input message="tns:SignInSoapIn" />
    <wsdl:output message="tns:SignInSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="SignOut">
    <wsdl:input message="tns:SignOutSoapIn" />
    <wsdl:output message="tns:SignOutSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="SignInMultiple">
    <wsdl:input message="tns:SignInMultipleSoapIn" />
    <wsdl:output message="tns:SignInMultipleSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="SignOutMultiple">
    <wsdl:input message="tns:SignOutMultipleSoapIn" />
    <wsdl:output message="tns:SignOutMultipleSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetActiveResponseGroups">
    <wsdl:input message="tns:GetActiveResponseGroupsSoapIn" />
    <wsdl:output message="tns:GetActiveResponseGroupsSoapOut" />
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ProxyServiceSoap" type="tns:ProxyServiceSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="IsAgent">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/IsAgent"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
```

```
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetAgent">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/GetAgent
" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetGroups">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/GetGroup
s" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SignIn">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/SignIn"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SignOut">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/SignOut"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SignInMultiple">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/SignInMu
ltiple" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SignOutMultiple">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/SignOutM
ultiple" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
```

```
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetActiveResponseGroups">
    <soap:operation
soapAction="http://schemas.microsoft.com/acd/2007/12/Microsoft.Rtc.Acd.Clients.Proxy/GetActiv
eResponseGroups" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>
```

7 Appendix B: Product Behavior

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

- Microsoft Lync Server 2013
- Microsoft Skype for Business (formerly Lync 2013)
- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Office Communications Server 2007 R2
- Microsoft Office Communicator 2007 R2
- Skype for Business
- Skype for Business Server

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

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

<1> Section 3.1.4.1: The **GetActiveResponseGroups** method (section 3.1.4.1) is not supported by Office Communicator 2007 R2 or Office Communications Server 2007 R2.

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

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

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

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

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

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

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

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

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

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

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

9 Index

T

V

W