[MS-PRES]:

Presence Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1		Initial version
4/25/2008	0.2		Updated based on feedback
6/27/2008	1.0		Updated and revised the technical content.
8/15/2008	1.01		Revised and edited the technical content.
12/12/2008	2.0		Updated and revised the technical content.
2/13/2009	2.01		Revised and edited the technical content.
3/13/2009	2.02		Revised and edited the technical content.
7/13/2009	2.03	Major	Revised and edited the technical content
8/28/2009	2.04	Editorial	Revised and edited the technical content
11/6/2009	2.05	Minor	Revised and edited the technical content
2/19/2010	2.06	Editorial	Revised and edited the technical content
3/31/2010	2.07	Major	Updated and revised the technical content
4/30/2010	2.08	Editorial	Revised and edited the technical content
6/7/2010	2.09	Editorial	Revised and edited the technical content
6/29/2010	2.10	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.10	No Change	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	3.0	Major	Significantly changed the technical content.
11/15/2010	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	4.0	Major	Significantly changed the technical content.
4/11/2012	4.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	4.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	4.1	Minor	Clarified the meaning of the technical content.
2/11/2013	4.2	Minor	Clarified the meaning of the technical content.

Date	Revision History	Revision Class	Comments
7/30/2013	4.3	Minor	Clarified the meaning of the technical content.
11/18/2013	4.3	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	5.0	Major	Significantly changed the technical content.
4/30/2014	5.1	Minor	Clarified the meaning of the technical content.
7/31/2014	5.2	Minor	Clarified the meaning of the technical content.
10/30/2014	5.2	No Change	No changes to the meaning, language, or formatting of the technical content.
3/30/2015	6.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

This document specifies the Presence Protocol, presence-related extensions to the Session Initiation Protocol (SIP). SIP is used by terminals to establish, modify, and terminate multimedia sessions or calls. The extensions that make up the Presence Protocol are used by the products listed in section 11 to allow users to provide presence-related data to other interested users over SIP.

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:

200 OK: A response to indicate that the request has succeeded.

403 Forbidden: A response that indicates that a protocol server understood but denies a request.

access control list (ACL): A list of access control entries (ACEs) that collectively describe the security rules for authorizing access to some resource; for example, an object or set of objects.

Active Directory: A general-purpose network **directory service**. **Active Directory** also refers to the Windows implementation of a **directory service**. **Active Directory** stores information about a variety of objects in the network. Importantly, user accounts, computer accounts, groups, and all related credential information used by the Windows implementation of Kerberos are stored in **Active Directory**. **Active Directory** is either deployed as Active Directory Domain Services (AD DS) or Active Directory Lightweight Directory Services (AD LDS). [MS-ADTS] describes both forms. For more information, see [MS-AUTHSOD] section 1.1.1.5.2, **Lightweight Directory Access Protocol (LDAP)** versions 2 and 3, Kerberos, and DNS.

activity: A type of event, such as "In a meeting," that provides information about the availability and status of a **presentity**.

aggregation: An operation in which multiple instances of one or more dependent **categories**, which are typically published by different Session Initiation Protocol (SIP) clients of the same user, are processed to produce an instance of a category. After this category instance is created, it can be published to multiple containers and notified to subscribers in the same way as any other category.

availability: A numerical value that indicates whether a user can be interrupted for communication. The higher the number, the less available the user.

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

Best Effort NOTIFY (BENOTIFY): A **Session Initiation Protocol (SIP)** method that is used to send notifications to a subscriber, as described in [MS-SIP]. Unlike the NOTIFY method, the BENOTIFY method does not require the recipient of the request to send a **SIP response**.

bot: A structured HTML comment that is processed by a front-end web server when the containing document is opened by or saved to the server. Also referred to as web bot.

category: An enhanced presence concept that is used by a **Session Initiation Protocol (SIP)** client to publish or subscribe to presence (2) information. A category enables basic identification of the data that is being published; it implies an agreed-upon schema for interpreting the data. A category name identifies a contract between a publisher and a subscriber.

category subscriber: A **SIP protocol client** that sent a **category SUBSCRIBE** request.

child element: In an **XML document**, an element that is subordinate to and is contained by another element, which is referred to as the parent element.

conference: A Real-Time Transport Protocol (RTP) session that includes more than one **participant**.

container: A data model that is used to store published presence (1) information and a list of subscribers who are permitted to view that information. It enables a publisher to publish different data values of the same **category** and instance, which enables different subscribers to see different values.

content type: A named and uniquely identifiable collection of settings and fields that store metadata for individual items in a SharePoint list. One or more content types can be associated with a list, which restricts the contents to items of those types.

Content-Type header: A message header field whose value describes the type of data that is in the body of the message.

Coordinated Universal Time (UTC): A high-precision atomic time standard that approximately tracks Universal Time (UT). It is the basis for legal, civil time all over the Earth. Time zones around the world are expressed as positive and negative offsets from UTC. In this role, it is also referred to as Zulu time (Z) and Greenwich Mean Time (GMT). In these specifications, all references to UTC refer to the time at UTC–0 (or GMT).

cyclic redundancy check (CRC): An algorithm used to produce a checksum (a small, fixed number of bits) against a block of data, such as a packet of network traffic or a block of a computer file. The CRC is used to detect errors after transmission or storage. A CRC is designed to catch random errors, as opposed to intentional errors. If errors might be introduced by a motivated and intelligent adversary, a cryptographic hash function should be used instead.

delegate: A user or resource that has permissions to act on behalf of another user or resource.

delegator: A user or resource for which another user or resource has permission to act on its behalf.

deployment: A collection of protocol clients and protocol **servers** that belong to the same enterprise.

dialog: A peer-to-peer **Session Initiation Protocol (SIP)** relationship that exists between two user agents and persists for a period of time. A dialog is established by **SIP messages**, such as a 2xx response to an INVITE request, and is identified by a call identifier, a local tag, and a remote tag.

directory service (DS): A service that stores and organizes information about a computer network's users and network shares, and that allows network administrators to manage users' access to the shares. See also **Active Directory**.

domain: A set of users and computers sharing a common namespace and management infrastructure. At least one computer member of the set must act as a domain controller (DC) and host a member list that identifies all members of the domain, as well as optionally hosting the **Active Directory** service. The domain controller provides authentication (2) of members, creating a unit of trust for its members. Each domain has an identifier that is shared among its members. For more information, see [MS-AUTHSOD] section 1.1.1.5 and [MS-ADTS].

endpoint: A device that is connected to a computer network.

enhanced presence: A presence model that uses **categories** to specify presence information and uses **containers** to authorize **subscribers**. This model includes **Session Initiation Protocol**

(SIP) extensions for publishing and subscribing to presence information and for specifying access control lists for subscribers. It uses the msrtc-event-categories presence document format.

event package: A specification that defines a set of state information to be reported by a notifying **Session Initiation Protocol (SIP)** client to a subscriber. An event package also defines further syntax and semantics based on the framework that is required to convey such state information.

federated user: An external user who possesses valid credentials with a federated partner and who therefore is treated as authenticated by a protocol server.

Globally Routable User Agent URI (GRUU): A **URI** that identifies a user agent and is globally routable. A URI possesses a GRUU property if it is useable by any **user agent client (UAC)** that is connected to the Internet, routable to a specific user agent instance, and long-lived.

hash: A fixed-size result that is obtained by applying a one-way mathematical function, which is sometimes referred to as a hash algorithm, to an arbitrary amount of data. If the input data changes, the hash also changes. The hash can be used in many operations, including authentication (2) and digital signing.

instance: A unique publication of data for a **category**. It enables a **publisher** to publish data for the same category multiple times. An example is a publisher who uses two different **endpoints** to publish data. These endpoints can publish the same category. However, each endpoint requires a different instance number to be considered a distinct publication by the **server**. An instance number is provided by the publishing client.

Lightweight Directory Access Protocol (LDAP): The primary access protocol for **Active Directory**. Lightweight Directory Access Protocol (LDAP) is an industry-standard protocol, established by the Internet Engineering Task Force (IETF), which allows users to query and update information in a **directory service (DS)**, as described in [MS-ADTS]. The Lightweight Directory Access Protocol can be either version 2 [RFC1777] or version 3 [RFC3377].

ms-diagnostics header: A header that is added to a **Session Initiation Protocol (SIP)** response, BYE request, or CANCEL request to convey troubleshooting information.

MSRTC: An abbreviation for Microsoft Real-Time Communications.

Multipurpose Internet Mail Extensions (MIME): A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], [RFC2046], and [RFC2047].

notification: A process in which a subscribing **Session Initiation Protocol (SIP)** client is notified of the state of a subscribed resource by sending a NOTIFY message to the subscriber.

NOTIFY: A method that is used to notify a **Session Initiation Protocol (SIP)** client that an event requested by an earlier SUBSCRIBE method has occurred. The notification optionally provides details about the event.

participant: A user who is participating in a **conference** or peer-to-peer call, or the object that is used to represent that user.

persistent SUBSCRIBE: A SUBSCRIBE request that is used to obtain updates when presence information changes for a **presentity**. The **subscription** does not expire until the subscribing client **endpoint** registration on a Session Initiation Protocol (SIP) server expires. See also **polling SUBSCRIBE**.

polling SUBSCRIBE: A **SUBSCRIBE** request that is used to obtain a one-time snapshot of presence information for a **presentity**. It has a value of "0" (zero) in the Expires header field and does not have any tag in the To header field.

Presence Information Data Format (PIDF): A common data format defined in [RFC3863] to exchange presence information.

presentity: An entity that provides presence information to a presence service.

public cloud (publicCloud) user: An external user who possesses valid credentials with a public, federated service, such as AOL, MSN, or Yahoo, and therefore is treated as authenticated by a **server**.

public switched telephone network (PSTN): Public switched telephone network is the voice-oriented public switched telephone network. It is circuit-switched, as opposed to the packet-switched networks.

publish: A SERVICE request that specifies which **category** instances to publish for a **presentity**.

publisher: A **SIP protocol client** that is making a publish request.

REGISTER: A **Session Initiation Protocol (SIP)** method that is used by an SIP client to register the client address with an SIP server.

same enterprise (sameEnterprise) user: An internal user who belongs to the same organization as another user who is sharing a communication session and is authenticated within that organization.

self SUBSCRIBE: A **SUBSCRIBE** request that is used by a **publisher** to be notified of changes to its own data. It is possible to subscribe to three different sets of data: **categories**, **containers**, and **subscribers**.

self subscriber: A **SIP protocol client** that is making a subscribe request for self-published **category** information.

server: A replicating machine that sends replicated files to a partner (client). The term "server" refers to the machine acting in response to requests from partners that want to receive replicated files.

SERVICE: A method that is defined by **Session Initiation Protocol (SIP)** extensions and is used by an SIP client to request a service from a server.

Session Initiation Protocol (SIP): An application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. **SIP** is defined in [RFC3261].

Simple Mail Transfer Protocol (SMTP): A member of the TCP/IP suite of protocols that is used to transport Internet messages, as described in [RFC5321].

SIP message: The data that is exchanged between **Session Initiation Protocol (SIP)** elements as part of the protocol. An SIP message is either a request or a response.

SIP protocol client: A network client that sends **Session Initiation Protocol (SIP)** requests and receives SIP responses. An SIP client does not necessarily interact directly with a human user. **User agent clients (UACs)** and proxies are SIP clients.

SIP request: A **Session Initiation Protocol (SIP)** message that is sent from a **user agent client (UAC)** to a **user agent server (UAS)** to call a specific operation.

SIP response: A **Session Initiation Protocol (SIP)** message that is sent from a **user agent server (UAS)** to a **user agent client (UAC)** to indicate the status of a request from the UAC to the UAS.

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].

SUBSCRIBE: A **Session Initiation Protocol (SIP)** method that is used to request asynchronous notification of an event or a set of events at a later time.

subscriber: A **Session Initiation Protocol (SIP)** client that is making a SUBSCRIBE request.

subscription: The result of a SUBSCRIBE request from a **Session Initiation Protocol (SIP)** element.

survivable mode: A mode that enables a protocol client to access basic voice services if some server or network resources are unavailable.

token: A word in an item or a search query that translates into a meaningful word or number in written text. A token is the smallest textual unit that can be matched in a search query. Examples include "cat", "AB14", or "42".

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [X509]). **TLS** is standardized in the IETF TLS working group. See [RFC4346].

tuple: An ordered grouping of members from different dimensions or hierarchies. A single member is a special case of a tuple and can be used as an expression. Every hierarchy does not have to be represented in a tuple.

Unified Communications: A system that integrates platforms for communications including email, voice mail, telephony, instant messaging, and voice and video conferencing.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

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

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

user agent client (UAC): A logical entity that creates a new request, and then uses the client transaction state machinery to send it. The role of **UAC** lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a **UAC** for the duration of that transaction. If it receives a request later, it assumes the role of a **user agent server (UAS)** for the processing of that transaction.

user agent server (UAS): A logical entity that generates a response to a **Session Initiation Protocol (SIP)** request. The response either accepts, rejects, or redirects the request. The role of the UAS lasts only for the duration of that transaction. If a process responds to a request, it acts as a UAS for that transaction. If it initiates a request later, it assumes the role of a **user agent client (UAC)** for that transaction.

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

XML document: A document object that is well formed, as described in [XML], and might be valid. An XML document has a logical structure that is composed of declarations, elements, comments, character references, and processing instructions. It also has a physical structure that is composed of entities, starting with the root, or document, entity.

XML element: An XML structure that typically consists of a start tag, an end tag, and the information between those tags. Elements can have attributes (1) and can contain other elements.

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.

[ISO-3166] International Organization for Standardization, "Codes for the representation of names of countries and their subdivisions -- Part1: Country codes", ISO 3166-1:2013, November 2013, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=63545

Note There is a charge to download the specification.

[ISO-8601] International Organization for Standardization, "Data Elements and Interchange Formats - Information Interchange - Representation of Dates and Times", ISO/IEC 8601:2004, December 2004, http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=40874&ICS1=1&ICS2=140&ICS3=30

Note There is a charge to download the specification.

[MS-OCER] Microsoft Corporation, "Client Error Reporting Protocol".

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

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

[MS-SIP] Microsoft Corporation, "Session Initiation Protocol Extensions".

[RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987, http://www.ietf.org/rfc/rfc1034.txt

[RFC1341] Borenstein, N., and Freed, N., "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, June 1992, http://www.rfc-editor.org/rfc/rfc1341.txt

[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

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", RFC 3261, June 2002, http://www.ietf.org/rfc/rfc3261.txt

[RFC3265] Roach, A. B., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002, http://www.ietf.org/rfc/rfc3265.txt

[RFC3863] Sugano, H., Fujimoto, S., Klyne, G., et al., "Presence Information Data Format (PIDF)", RFC 3863, August 2004, http://www.ietf.org/rfc/rfc3863.txt

[RFC4480] Schulzrinne, H., Gurbani, V., Kyzivat, P., and Rosenberg, J., "RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF)", RFC 4480, July 2006, http://www.rfc-editor.org/rfc/rfc4480.txt

[SOAP1.1-Envelope] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1 Envelope", May 2001, http://schemas.xmlsoap.org/soap/envelope/

1.2.2 Informative References

[IETFDRAFT-SIPSOAP-00] Deason, N., "SIP and SOAP", draft-deason-sip-soap-00, June 30 2000, http://www.softarmor.com/wgdb/docs/draft-deason-sip-soap-00.txt

[MS-CONFBAS] Microsoft Corporation, "Centralized Conference Control Protocol: Basic Architecture and Signaling".

[MS-SEARCH] Microsoft Corporation, "Search Protocol".

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

[MS-XCCOSIP] Microsoft Corporation, "Extensible Chat Control Over Session Initiation Protocol (SIP)".

[RFC4482] Schulzrinne, H., "CIPID: Contact Information for the Presence Information Data Format", RFC 4482, July 2006, http://www.rfc-editor.org/rfc/rfc4482.txt

1.3 Overview

This protocol is an extension of the original **Session Initiation Protocol (SIP)**, as described in [RFC3261].

This protocol adds a number of extensions to presence, as described in [RFC3261], [RFC3265], and [RFC3863], in addition to [MS-SIP]. The **Presence Information Data Format (PIDF)**, called msrtc.pidf, is described in [MS-SIP]. This document introduces a newer presence model called **enhanced presence**. The presence document format is named msrtc-event-categories. Enhanced presence involves a presence data exchange model and a way to authorize subscribers for presence data over SIP. Enhanced presence-related extensions to SIP are briefly described in the following paragraphs, and are described further in section 3.

Note that enhanced presence is not related to the protocol described in [RFC4480], which defines rich presence extensions to the presence information data format.

This protocol includes protocol enhancements for publishing and subscribing to presence-related data, as well as authorizing **subscriber** access to **publishers'** presence-related data. The Enhanced Presence Model allows a user, also known as a **presentity** (presence entity), to exercise finer control in publishing presence information. It introduces a number of new concepts, such as the publications described in section 1.3.1.1, categories, and **containers**. Users **publish** different categories of data to the **server**, rather than a single document, as in msrtc.pidf, and subscribers subscribe to different categories published by the user. This protocol allows arbitrary categories to be exchanged between the publisher and the subscriber.

Containers provide the publisher with a way to publish different data values of the same **category** for the purpose of allowing different subscribers to see different values. This gives the publisher more granular control compared to the **access control list (ACL)** model described in [MS-SIP]. Extensions defined in this document allow publishing data, managing containers, and subscribing to these categories by sending XML requests carried within the body of **SERVICE** and **SUBSCRIBE** requests.

Enhanced presence **subscription** extensions make use of certain concepts and optimizations previously introduced by [MS-SIP], such as Piggyback **NOTIFY** and **Best Effort NOTIFY (BENOTIFY)**. The extensions also use a similar batched SUBSCRIBE request to reduce the number of subscription and **notification** messages exchanged between the **SIP protocol client** and the server. Note that BENOTIFY and NOTIFY samples are used interchangeably in various sections that explain the Presence Protocol-related XML elements and attributes, because the content is independent of the NOTIFY type.

The architecture for the Enhanced Presence Model is described in detail in section 1.3.1.

This protocol also specifies the SIP-based **directory service (DS)** search for users to be able to locate other users to communicate with. For details about SIP-based DS search, see section 1.3.2.

This protocol also specifies how the protocol client discovers the site server search service's **Uniform Resource Locator (URL)**. The site server search service is described in [MS-SEARCH].

This protocol also specifies the support for PIDF/RPID and **MSRTC** presence document format in the server. For more details on PIDF/RPID and MSRTC support, see section 1.3.3.

This protocol also specifies the **delegate** management operations on the server. For more details on delegate management, see section 1.3.4.

This protocol also specifies the support for the Unified Contact Store (UCS) for both the client and server. For more details on the UCS feature, see section 1.3.5.

This protocol also specifies the support for persistent chat systems based on the [MS-XCCOSIP] protocol. For more details on this feature see section 1.3.6.

1.3.1 Enhanced Presence Architecture

This section provides detailed information about the following elements of the enhanced presence model as defined by this protocol:

- Publishing categories.
- Container semantics.
- Subscriber list management.
- **Self SUBSCRIBE**.
- Versioning semantics.
- Aggregating multiple **instances**.
- Container management.

1.3.1.1 Publishing Categories

A SIP protocol client can create or update a **category** instance and send a request to a server to make that new or updated **category** instance available to subscribing users. This process is known as publishing a category and requires that a client generate and send a SERVICE request carrying the item, or **category** instance, to be published.

In the enhanced presence model, publishers provide the following information when they create or update a published item:

- Publisher identity.
- Container number.
- Category name.
- Instance number.
- Version number.
- Expire type.
- Expires (optional).
- Data value.

The combination of publisher identity, container number, category name, and instance number uniquely identifies a publication in the server.

Following is an overview of these concepts:

Publisher identity: A publisher identity establishes the identity of the user who generates a publication and publish request. A published item is always an aspect of the presence of the publishing user. A SIP publish request carries the publisher identity in the form of a SIP **Uniform Resource Identifier (URI)** formatted string. The publisher's identity is also encoded within the XML body of a publish request using the same SIP URI as is used in the previously mentioned SIP headers.

Container number: A container number is the unique identifier of a container. The container number assigned to a published item specifies a single specific container in a collection of containers into which a **category** instance is to be published.

Containers allow the publisher to publish different data values of the same category and instance to different subscribers. For example, as part of presence, a publisher might intend to show a personal note to one class of subscribers but a more generic note to another class of subscribers. To do this, the publisher publishes the "note" category twice. Each publication targets a different container with a different value.

Containers hold a set of published items and a list of members specifying who can see those items. The membership list is defined and maintained by the publisher independently from the published items.

Category name: Categories identify what data is being published. A **category name** is an identifier of the contract between a publisher and a subscriber. It identifies the nature of the published data, and it implies an agreed-upon schema for interpreting the data.

Instance number: Instances allow the same publisher to publish the same category of data multiple times. For example, a publisher might use two different **endpoints** to publish data. These endpoints might publish the same category and yet be treated by the server as distinct publications. In that case, each publication needs a different instance number. Instance numbers are provided by the publishing SIP protocol client.

Version number: Each published item has a server-maintained version number. This number is provided so that multiple SIP protocol clients that publish the same instance have a way to stay synchronized. A publication is updated only when a publisher presents the current version number. The version number is then incremented automatically with the data update by the server.

Expire type: The lifetime of a publication can be bounded in any of the following ways:

- Endpoint-bounded: The publication lives as long as the publishing endpoint stays registered.
- User-bounded: The publication lives as long as the publishing user stays registered from at least one endpoint.
- Time-bounded: The publication expires if it is not refreshed within a certain period of time.
- Unbounded (static): The publication never expires.

Expires: Time-bounded publications have a SIP protocol client-specified lifetime. Each time the published item is updated, the expiration time is updated with the new value provided by the SIP protocol client. If the published item is not subsequently updated, the server automatically deletes it after the expiration time is reached. This value is optional.

Data value: The actual enhanced presence information carried within a **category** instance. The data value string is an **XML document** conforming to an enhanced presence schema document. The conformity of the XML document is enforced by the server.

In addition to the preceding attributes, the server also maintains the last time an item was published, referred to as last publish time, and makes this information available in subsequent notifications about the item. This data can be used by subscribers in some situations to resolve any ambiguities over whether a change is newer than the data they already have for the item. The granularity of the last published time is only to the minute.

1.3.1.2 Private Categories

In the enhanced presence model, the server can support registering a category as a "private" category. The server does not allow subscription to private categories from subscribers. The server allows publication of private categories into any container. Some of the categories, like routing (section 2.2.2.7.7) and mwi (section 2.2.2.7.11), can be marked as private categories to not allow subscriptions for those from subscribers. Based on this, Private categories can be present in any container and visible only for the **self-user**.

1.3.1.3 Container Semantics

The following sections specify the container semantics for the enhanced presence architecture.

1.3.1.3.1 Access Control to Published Enhanced Presence Data

Access control to published enhanced presence is supported using containers. An enhanced presence container holds a list of presence data and a list of subscribers permitted to receive the contained, or published, presence data. A container thus prescribes a mapping to determine who can access which presence information, and it functions very much the same as an ACL. Containers provide a flexible authorization model for a user to control the amount of presence information that can be accessed by others.

A container is created and maintained by the server. The access level semantics, or access control, for containers, with the exception of container 0, are determined and can be modified by a SIP protocol client.

Containers are identified by an unsigned short integer. Container 0 is assigned special status as a default container.

A container holds a set of published items and a membership list describing who can see those items. Container membership is defined in section 1.3.1.3.5.

1.3.1.3.2 Granting Access to Publications

By manipulating a container's membership, the publisher can allow or prevent subscribers from seeing the published items in that container. A subscriber who is in the membership list of the container is allowed to see the categories published to that container. Conversely, a subscriber who is not in the membership list is not allowed to see the publications of that container.

1.3.1.3.3 Container 0: The Default Container

Publisher-defined membership lists control access to every container except container 0, the default container. By default, a subscriber is allowed access to every category published in container 0. This convention makes it easy for protocol clients to publish default values for all categories that are available to anyone. Essentially, the default container contains only publications and has no membership list. Modifying the membership list of container 0 is not allowed.

1.3.1.3.4 Blocking Access to Publications

A subscriber is blocked from seeing a category when the subscriber is not a member of any container that publishes that category. In other words, a subscriber is blocked when the intersection of all containers that contain the published category and all containers that contain the subscriber in their membership lists result in an empty set.

When a subscriber is blocked from seeing a category, the subscriber sees an empty data value. In this way, the subscriber cannot tell the difference between a publisher who has not published the given category and a publisher who has blocked the subscriber.

If the presence of an empty value for a category could imply to the subscriber that it has been blocked, the publisher can instead publish a data value that has plausible, but not very useful, data for the category to the default container. This is another reason that the default container has no membership list.

1.3.1.3.5 Container Membership

A subscribing user's membership in a container is expressed as the union of the following optional items:

- A list of specific subscriber URIs.
- A list of SIP **domains**.
- All **same enterprise (sameEnterprise) users**.
- All **federated users**, which are referred to as federated.
- All **public cloud (publicCloud) users**.
- Everyone.

Note that the first and second items are lists of items, whereas **federated**, **sameEnterprise**, **publicCloud**, and **Everyone** are Boolean attributes of the container.

1.3.1.3.6 How a Subscriber Is Resolved to a Container

When a subscriber subscribes to a category that is published in multiple containers, the question of which container's value to use needs to be answered.

Only those containers that have the given published category are considered.

If there is only one container that has the subscriber as a member, there is no ambiguity, and that container is used to satisfy the subscription.

When multiple containers have the subscriber in their membership, the subscription is satisfied from the highest-numbered container. This is where ordering of containers based on their number applies. When containers have overlapping membership, containers with higher numbers take precedence over those with lower numbers.

1.3.1.3.7 Evaluating Membership

A subscriber can be contained in the membership of more than one container in different ways. For example, one container can specify "friend@aol.com" in its URI list, while another container specifies "publicCloud". In this case, "friend@aol.com" is considered a member of both containers.

Similarly for example, among alice@contoso.com's containers, one container can specify "sameEnterprise", which contains bob@contoso.com, while another container specifies "contoso.com". In this case, bob@contoso.com is a member of both containers.

In these situations, the subscription is matched in the following priority order:

1. The URI list.
2. The domain list.
3. A user in the same enterprise.
4. A federated user.
5. A public cloud user.
6. Everyone.

See section 3.2.5.3 for more information on resolving a subscriber to a container. If no container is found, the subscriber is blocked.

1.3.1.3.8 Example Containers

The following figure shows sample containers for alice@contoso.com.

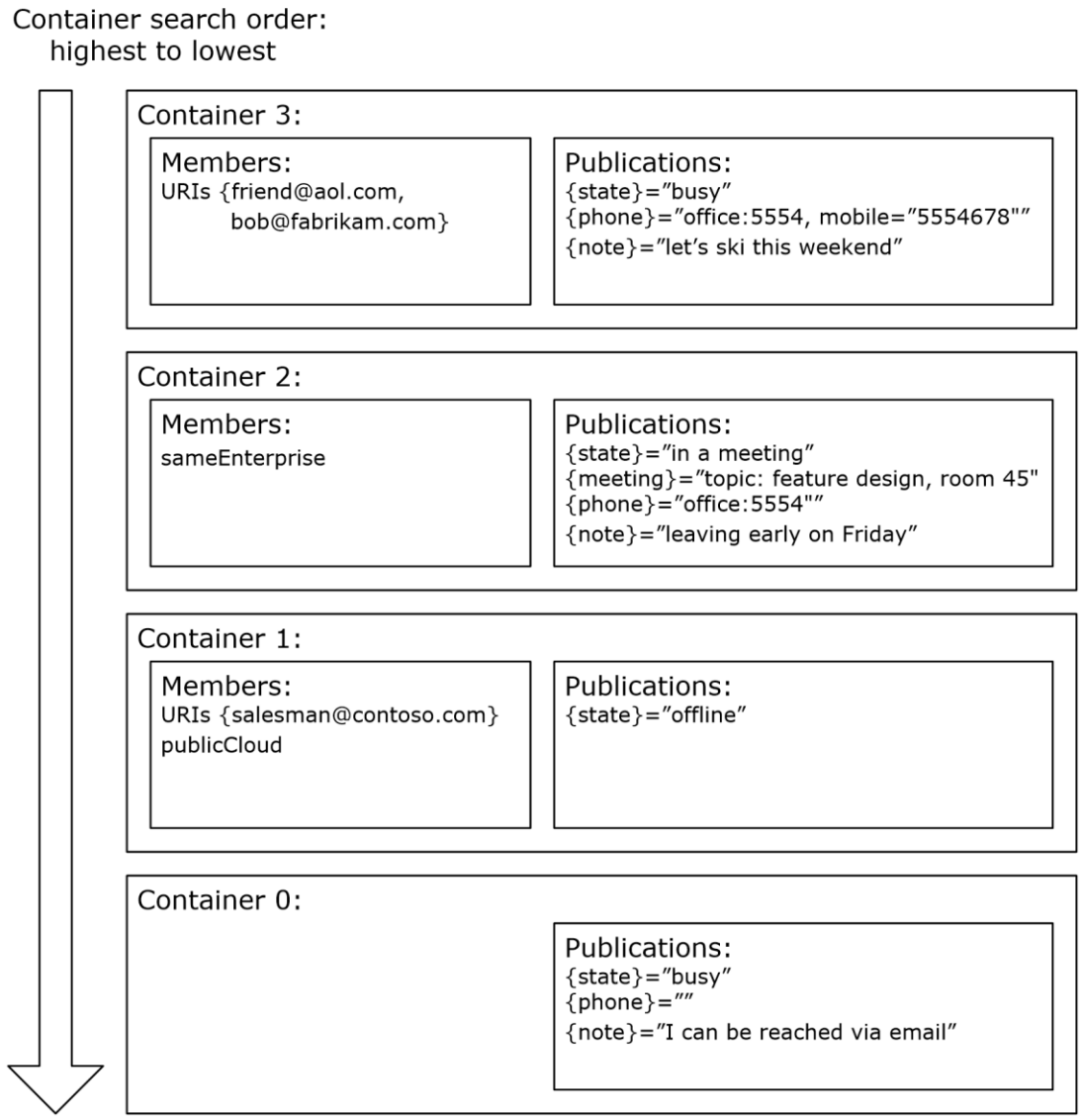

Figure 1: Container search order

Container 0 shows default publications that are available to anyone who does not fall into the membership of higher containers. For example, federated users are not specified anywhere else, so those users other than "bob@fabrikam.com" get data from this default container.

Container 1 has been set up to give one person and all public cloud subscribers a blocked view of state. "Salesman@contoso.com" gets phone, meeting, and note categories from container 2, because they are in the same domain as the publisher. If the publisher does not intend for these categories to be given to the blocked "contoso.com" user, the categories are also published to container 1.

Container 2 is the set of categories published to people in the same enterprise as the publisher, unless those people are satisfied by a higher container or a lower container with a more restrictive membership, such as being on a URI list.

Container 3 shows a more detailed view of phone and note information for the specific people in the URI list.

1.3.1.3.9 Re-Resolution When Publications Are Added or Removed

The first time a category is added to a container, as opposed to being subsequently updated, all existing subscriptions to that category need to be reevaluated to determine whether the subscription needs to be relocated to this container.

Similarly, when a publication for a category is removed from a container, all subscriptions to that category need to be reevaluated to determine whether the subscriptions need to be relocated to a different container.

1.3.1.3.10 Operations That Require Re-resolution

This section contains a list of operations that require the server to reevaluate existing category subscriptions to see whether there is a need to notify subscribers to the publisher's categories:

- Publication of a new category.
- Modification or deletion of an existing publication.
- Changes in publisher's container memberships.
- All cases that cause one of a publisher's endpoints to be de-registered:
- The user sends a **REGISTER** with **Expires** set to zero.
- The server terminates a publisher's endpoint because of inactivity.
- The server takes the publisher offline for administrative purposes, such as server maintenance.
- The server deletes time-bounded publications.
- The server publishes a category on behalf of a user. For more information, see section 3.2.6.2.

Note that some of these cases are implementation-dependent and might not apply to all server implementations.

1.3.1.4 Subscriber List Management

In the enhanced presence model, when a publisher receives a new subscription, the server returns the data according to the publisher's containers. The publisher is not informed of the new subscription unless there is a **context** element in the body of the subscription XML. The subscriber list is the mechanism used to inform a publisher about new subscribers.

The subscriber list serves two main purposes:

- Level of access: SIP protocol clients typically set up containers to give subscribers a minimum, default level of access. Subscribers can use this mechanism to notify publishers, which can cause SIP protocol clients to prompt the publisher to determine the correct level of presence data to be returned to this subscriber.
- Social: This is the most common way users add each other to their contact lists.

The server maintains a list of subscriber URIs. Each item in this list has a Boolean attribute, **acknowledged**. The semantics for this list are as follows:

- After an enhanced presence subscription with a **context** element is received and processed normally, this list is searched for an entry that matches the URI of the subscriber. If one is found, no further action is taken.

- If an entry is not found, a new entry is added for the subscriber's URI with the **acknowledged** attribute's value set to **false**.

1. A notification is sent to the publisher containing all URIs in this list with the **acknowledged** attribute value set to **false**. This notification is sent using the self SUBSCRIBE **dialog**, which is discussed in section 1.3.1.5.

2. The SIP protocol client then informs the server to set the **acknowledged** attribute to **true** for one or more URIs in this list.

The **acknowledged** attribute is needed to ensure that notification of the new subscribers happens reliably at least once and can be roamed to other endpoints belonging to the publisher for deferral of the acknowledgment until a later time.

These semantics also imply that the publisher will subscribe to this list through self SUBSCRIBE and receive notifications when the list changes. For simplicity, only full state change notifications are supported in the notifications. That is, all entries in the subscriber list are present in notifications, and not just the ones that changed.

The SIP protocol client can send a **setSubscriber** request to set the **acknowledged** attribute associated with the subscriber to true.

The server can remove the subscriber entry from the subscriber list after receiving acknowledgement for it from the SIP protocol client. This is to avoid enlarging the list of subscribers stored on the server.

For details about subscriber list management, see section 2.2.2.6 and section 3.6.

1.3.1.5 Self SUBSCRIBE

Publishers need the ability to subscribe to their own data and to be notified when there are changes. Self SUBSCRIBE is a single SIP SUBSCRIBE dialog that is used for this purpose. It is possible to subscribe to four different sets of data through this dialog:

- **Categories:** A publisher is notified of changes to the published data in its containers when the publisher is subscribed to self category data. Each notification contains the container number, changed category, instance number, version, expiration information, and the data of the changed or new publication. The initial NOTIFY or the piggyback **200 OK** contains all publications of the user from all containers. Subsequent change notifications include all the publications of the same category from containers that were affected by the change, such as batch publications. When the last publication of a category in a container is deleted, an empty notification is generated.

- **Containers:** Publishers are notified of changes to the membership of their containers when they are subscribed to "containers". Each notification contains the identity of the container whose membership has changed, along with the complete new membership of the container. The initial notification contains all membership information for all containers. Subsequent change notifications include all the members of containers that were affected by the change, such as batch **setContainerMember** operations.

- **Subscribers:** Publishers are notified of changes to their subscriber list when they are subscribed to "subscribers". Each notification contains the full subscriber list.

- **Delegates:** Publishers are notified of changes to their delegate list when they are subscribed to "delegates". Each notification contains the full delegate list in addition to the current version of this document.

These sets of data define the scope of a self subscription dialog. The self SUBSCRIBE **SIP request** has an XML body that allows the SIP protocol client to modify the scope. Details of the request and notifications are described in section 2.2.2.3 and section 3.3.

1.3.1.6 Versioning Semantics

Both the publication version number and the last publish time are used to resolve whether a change notification represents the latest change, which is normally accepted, or an older change, which is normally ignored. Note that version information is made available only in notifications to **self subscribers**, whereas the last publish time is available in notifications to both self subscribers and subscriptions received from other users.

1.3.1.6.1 Creating a Published Instance

To publish an instance of a category for the first time, or create the instance, the publisher provides a version number equal to zero in the request. The server creates the instance and set its version number to one only if the instance does not already exist.

If the item already exists, the server fails the request and returns information about the existing item, including its current version number, its last publish time, and the data itself as specified in section 2.2.2.2.3. The publisher normally then processes this returned information.

On a successful creation, the last publish time is returned to the publisher, and notifications of all instances of this category are sent to all subscribers.

1.3.1.6.2 Updating a Published Instance

Publishers present the current version number when they request that an item be updated. The server updates the instance and increments its version number only if the publisher-provided version number matches the current version.

If the version does not match, the server fails the request and returns information regarding the item, including its current version number, its last publish time, and the data itself, as described in section 2.2.2.2.3. The publisher normally then processes this returned information.

On a successful update, the last published time is returned to the publisher and notifications of all instances of this category is sent to all subscribers to this category.

1.3.1.6.3 Deleting a Published Instance

To delete a publication, or instance of a category, the publisher presents the current version number when it requests the deletion. The server accepts the request only if the provided version matches the stored version on the server.

If the version does not match, the server fails the request and returns information about the item including its current version number, its last publish time, and the data itself, as described in section 2.2.2.2.3. The publisher normally then processes this returned information.

On a successful deletion, the time of the deletion is returned to the publisher and notifications of all instances, except the one deleted, of this category are sent to subscribers. The notification includes all instances of the subscribed-to category. The absence of the deleted instance in this notification tells the subscriber that the instance is no longer valid.

1.3.1.6.4 Re-creating a Published Instance

After an instance of a category has been deleted, it can be re-created.

The only note of interest for this case is the situation in which a subscriber misses the deletion notification and then receives the re-creation notification. The re-creation self notification indicates that the instance has a version of one, which is less than the version of the SIP protocol client, and a last publish time greater than the SIP protocol client. In this case, the SIP protocol client still accepts this update if the last publish time of the new version is more recent than that of the SIP protocol client's current version.

1.3.1.7 Aggregating Multiple Instances

Aggregation is needed to let multiple SIP protocol clients, devices, endpoints, or services, all publishing on behalf of the same user, arrive at a single unified value for an arbitrary category. These different publishing SIP protocol clients of the user are not generally aware of one another, nor are they aware of the different data that the others publish.

For example, a service might publish a user's calendar state, while the user's physical device might publish its **activity** as a device state. Both of these publications affect the user's overall presence status, but neither the service nor the device has all of the information that it needs to publish that overall status. The implication is that another processing entity, with access to both pieces of information and the logic to combine them, is necessary to produce the overall state. This other processing entity is chosen to be the server. The details of aggregation are described in section 3.8.

1.3.1.8 Container Management

Creation of a container is not an explicit operation. The server creates containers automatically when either a member is added or a publication is made.

1.3.1.8.1 Modifying Container Membership

Each container has a separate version number. This number is used to synchronize modifications to the membership of a container from multiple SIP protocol clients. The current version of the container membership is provided as input to any operation, such as addition or removal, that modifies the membership. If the SIP protocol client-provided version number matches the value on the server, the server increments the version number automatically with the modification. If the version number does not match, the current version and the membership data are returned. Attempting to delete a container member that does not exist does not cause a failure. Similarly, adding a container member that already exists is considered a modification and does not cause a failure.

For details about container management, see section 2.2.2.5 and section 3.5.

1.3.2 SIP-Based Active Directory Search

Users need to be able to locate other users to communicate with. This is particularly the case in corporate scenarios where there are a large number of users with whom a user potentially communicates. The user can search using a SIP SERVICE request. The server receives the request and executes an ambiguous name resolution search against the DS. The result set is then passed back to the SIP protocol client as the **Simple Object Access Protocol (SOAP)** body of a 200 OK response. The user then picks who to communicate with from the result set. The SIP protocol client can include only attributes of a person that are available in the DS global catalog. The server can apply additional filters according to the policy specified by an administrator. The server returns information only on users that the requester has rights to see in a DS. The administrator can limit the number of users that can be returned in a search.

1.3.3 PIDF/RPID and MSRTC Support

The server is required to support incoming subscriptions to three types of presence document formats; however, the order of preference that follows is to be used if the subscriber indicates support for all three:

- Enhanced presence (**Content-Type:** application/msrtc-event-categories+xml).
- MSRTC (**Content-Type:** text/xml+msrtc.pidf).
- PIDF (**Content-Type:** application/pidf+xml).

That is, for every incoming dialog-creating SUBSCRIBE request, the server searches the **Accept** header for these presence document formats in this order. When a match is found, the SUBSCRIBE request is terminated using the matched format. Subsequent SUBSCRIBE refresh requests indicate support for the same presence document formats.

The server supports the generation of PIDF documents for SIP subscription. The server does not support the PIDF and Rich Presence Extensions to the Presence Information Data Format (RPID), as described in [RFC4480], data formats in their entirety, but only the subset required for interoperability with federated presence clouds.

The PIDF document contains a single PIDF **tuple** that contains attributes from basic PIDF, as well as the following attributes from the RPID and Contact Information in PIDF (CIPID), as described in [RFC4482], extensions:

- **Available/activity:** Activity is an RPID extension that allows SIP protocol clients that provide presence information, called presentities, to attach information about what the **presentity** is actually doing.
- **display-name:** A CIPID attribute that provides the friendly name of the user.

The server passes these attributes in a single PIDF tuple.

The PIDF document can also contain a list of linked-identities of the user. The linked identities let the consumers know of all the URIs that could represent the same presentity. The cid field contains a unique contact ID for the presentity.

In addition to PIDF, the server supports MSRTC subscriptions. For details about the MSRTC presence document format, see [MS-SIP].

1.3.4 Delegate Management

Delegates are added and removed explicitly by a SIP protocol client using a SIP SERVICE request. A user whose **SIP:Uri** is in the delegate list is enabled to do certain operations on behalf of the delegator. This is further explained in [MS-SIPAE] section 3.2.4.4. These operations include the ability to do the following:

- Initiate outbound calls on behalf of the delegator.
- Receive inbound calls that are meant for the delegator. This is further explained in [MS-SIPAE].
- Join a **conference** on-behalf-of the delegator. This is further explained in [MS-CONFBAS].

A delegator is a user who submits a **setDelegate** request, as specified in section 2.2.2.10.2, to give another user permission to perform the previous operations.

1.3.5 Unified Contact Store

Unified Contact Store (UCS) is a feature through which the contacts can be migrated from the enhanced presence server's contact store to an email server contact store. This allows the contacts to be shared with other non-SIP enabled end points as well. There are three stages in the process of migration, start the migration, the period of migrating the contacts, completion of migration:

1. Start the migration: This can be done by a SIP protocol client using a SIP SUBSCRIBE for the contact list. It will let the enhanced presence server know that the client is ready to move its contacts to the email server. The client can only send this if it has connection to the email server.

2. The period of migrating the contacts: The contact and group list is put in a read only state while migration. All SERVICE messages meant to modify the contact list during this point will be rejected.

3. Completion of migration: Once the contacts have been migrated to the email server contact store, the messaging client will directly connect to the email server to retrieve the contact/group list. The messaging client can perform the same contact/group list operations it did on the email server as it can on the messaging server. The operations that the messaging client can do on the Contacts and Group List on the messaging server are detailed in [MS-SIP] section 3.7.4

   After migration, SIP End points that do not connect to the email server will connect to the enhanced Presence Server as documented in [MS-SIP] section 3.7.5.2. The enhanced presence server will then proxy the requests and responses to and from the email server.

1.3.6 Persistent Chat

The enhanced presence model can also be used for managing user preferences in a persistent chat messaging system built on [MS-XCCOSIP] protocol. A SIP protocol client can publish and self-subscribe to the following categories:

- roomSetting
- roomUpdate
- roomInvitation
- gcFilterSetting

The detailed content of these categories is described in section 2.2.2.7.13

1.4 Relationship to Other Protocols

This protocol depends on the Session Initiation Protocol (SIP). The SIP extensions, described in section 3 in [MS-SIP], define additional SIP primitives and XML schemas to support various extensions specified in this document.

This protocol is invoked as an extension of SIP. This protocol depends on all the protocols on which the SIP specification depends.

1.5 Prerequisites/Preconditions

This protocol assumes that both SIP protocol clients and the server support SIP. The prerequisites for SIP Extensions are the same as the prerequisites for SIP.

1.6 Applicability Statement

This protocol is applicable when both SIP protocol clients and the server support SIP and intend to use one or more of the enhancements offered by this protocol.

1.7 Versioning and Capability Negotiation

SIP extensions do not have protocol versioning. Instead, explicit capability negotiation is done as specified in this section by using the **Supported** header field to indicate support of various features. Using the **Supported** header field is the standard SIP mechanism of performing capability negotiation.

1.8 Vendor-Extensible Fields

Use standard SIP extension mechanisms as needed.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

These extensions to SIP do not introduce a new transport to exchange messages but are capable of being used with any transport used by SIP. **SIP message**s are transported over TCP or **Transport Layer Security (TLS)**.

2.2 Message Syntax

These extensions to SIP rely on the SIP message format, as specified in [RFC3261] section 7, and they extend definitions of the URI and header field parameters by adding new values for parameter and header names, as well as their corresponding values. This protocol also defines new message body types, in addition to those defined in [RFC3261] section 7.4.1. Header fields extended by this protocol in addition to those specified in [MS-SIPRE] include:

- SERVICE request and response
  - Content-Type
- 409 Conflict response
  - ms-diagnostics
- SUBSCRIBE request and response
  - Content-Type
  - Require

New message body types specified in this document are the following:

- Categories document
- Publish document
- Self-subscribe request
- Self-subscribe response
- Category subscribe request
- Category subscribe response
- Category publish request
- Containers document
- **setContainerMember** request
- Subscriber list document
- **setSubscriber** request
- **subscriptionContext**
- State category
- Device category

- Services category
- **userProperties** category
- **contactCard** category
- **legacyInterop** category
- **routing** category
- **calendarData** category
- **workingHours** category,
- **dndState** category,
- **mwi** category,
- **linkedPICContacts** category
- Delegates document
- **setDelegate** request

2.2.1 SIP-Based Active Directory Search

This section describes behaviors supported in versions described by endnote <1>.

A user uses a SERVICE request to perform a directory service (DS) user search. This protocol uses SIP requests and responses, as specified in [RFC3261] section 7.1 and section 7.2, as the carrier protocol. Specifically, the SIP SERVICE request and its corresponding **SIP response** as defined in [IETFDRAFT-SIPSOAP-00] are used for transmitting SIP messages.

All requests MUST be specified as the body of a SIP SERVICE request. All responses MUST be specified as the body of a SIP SERVICE response. The body MUST be well-formed XML and MUST be encoded using UTF-8. The XML MUST conform to the XML schemas in section 8.

In the following example, the user bob@contoso.com issues a DS user search against the SIP domain contoso.com and searches for all users whose name begins with "Alice".

```
SERVICE sip:contoso.com SIP/2.0
Via: SIP/2.0/TLS 192.168.66.91:49541
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=0d18c6dde0;epid=0199305c6d
To: <sip:contoso.com>
Call-ID: 717ff9c645de415688683fb482527aa0
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:E4abE0hR5VCdvhGcA7shLQAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service",
opaque="B1E794AB", crand="a9c657de", cnum="354", targetname="server.contoso.com",
response="01000000690070003bf4c8c33d9c49c2"
Content-Type: application/SOAP+xml
Content-Length: …

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <m:directorySearch xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip">
      <m:filter m:href="#searchArray"/>
      <m:maxResults>100</m:maxResults>
    </m:directorySearch>
    <m:Array xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip"
m:id="searchArray">
```

```
      <m:row m:attrib="givenName" m:value="Alice"/>
    </m:Array>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Elements and attributes available for a DS user search are defined as follows.

filter:href This attribute points to the element that contains the array of attributes to use for filtering search results returned from the DS. The element that contains the results will contain an **id** attribute that has a value equal to the value of the href.

maxResults: The maximum number of search results requested by the SIP protocol client.

Array: The filters for a search operation. Filters can be any DS attribute on a user object and can appear in any order in the array.

2.2.2 Enhanced Presence Messages

2.2.2.1 categories XML Document

A number of enhanced presence SIP messages carry an XML body that includes a list of **category** instances. The following example illustrates a categories XML document conforming to the schema defined in section 6.2. The categories XML document is used to provide a list of **category** instances.

```
<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
  <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="200" version="1" expireType="static">
    <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
      <body type="personal" uri="">Working until 5pm today</body>
    </note>
  </category>
  <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="300" version="1" expireType="static">
    <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
      <body type="personal" uri="">Working until 5pm today</body>
    </note>
  </category>
  <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="400" version="1" expireType="static">
    <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
      <body type="personal" uri="">Working until 5pm today</body>
    </note>
  </category>
</categories>
```

The **XML elements** of a categories document MUST conform to the schema defined in section 6.2.

categories: This element is a wrapper for a list of **category** instances. It has the following attribute:

> **uri (required):** This attribute specifies the publisher's SIP URI. All of the **category** instances included in the list are from the same publisher.

category: Each **category** element describes a unique **category** instance. It has the following attributes:

- **name (required):** This string value identifies the **category** to which the instance belongs.
- **instance (optional):** This unsigned integer specifies the instance number of the **category** instance. The publisher specifies an integer as an instance number to publish multiple instances of

the same category simultaneously in the same container. If no instance is specified, instance zero is assumed.

- **publishTime (optional):** This string value specifies the time, in **UTC**, when this **category** instance was last published. This time is maintained by the server and MUST have a granularity no greater than one minute. This value is used by self subscribers to determine whether to accept a notification.
- **container (optional):** This unsigned short integer specifies the container to which the **category** instance belongs**.** Publishers determine the set of **category** instances published to a container. The attribute MUST be present in all self-subscribe responses and publication responses and MUST NOT be present for category subscription responses.
- **version (optional):** This unsigned integer specifies the version number of the **category** instance maintained by the server.
- **expireType (optional):** This string value determines how the lifetime of the **category** instance is bound. The following values are defined and MUST be supported by the server based on the following description:
  - **static:** The **category** instance lives until explicitly deleted.
  - **endpoint:** The **category** instance lives as long as the publishing endpoint is registered.
  - **user:** The **category** instance lives as long as the publishing user has at least one registered endpoint.
  - **time:** The **category** instance lives until the specified expiration time. The expiration time can be updated in subsequent messages.

    The attribute MUST be present in all self-subscribe responses and publication responses and MUST NOT be present for category subscription responses.
- **endpointId (optional):** This string value identifies the endpoint that published this **category** instance. This attribute MUST be added to the document only for **category** instances where **expireType** has the value "endpoint". The value of **endpointId** is the **UUID** in the +sip.instance parameter value from the **Contact** header field according to syntax specified in [MS-SIPRE] section 2.2.5.
- **expires (optional):** This string value specifies the time, in seconds, after which this **category** instance will expire. This attribute MUST be specified for **category** instances where **expireType** has the value "time". It is also specified to indicate that a **category** instance is being deleted, and MUST then be set to zero.

The body of the **category** element includes the data returned to a client by a server in a response. This data SHOULD be interpreted according to the schema identified by the **name** attribute.

2.2.2.2 Publication-Related Messages

The following sections specify publication-related messages for enhanced presence.

2.2.2.2.1 publish Request

A publisher uses a SERVICE request to publish **category** instances. The following example illustrates the format of such a message.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=b5410171e2;epid=84d3db8c23
```

```
To: <sip:bob@contoso.com>
Call-ID: 82369a8c95ba4778b3b9220e4abb73d8
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu="">
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-category-publish+xml
Content-Length: …

<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="note" instance="0" container="300" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
    <publication categoryName="note" instance="0" container="400" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
    <publication categoryName="note" instance="0" container="200" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
  </publications>
</publish>
```

The **To** and **From** header fields in a **publish** request MUST carry the same SIP URI, that of the publisher. The **Content-Type** header field MUST be "application/msrtc-category-publish+xml", which means that this SERVICE carries an XML body that contains a list of **category** instances to be published.

Elements and attributes for a category publish request are defined as follows.

The XML elements of a publish document MUST conform to the schema defined in section 6.10.

publish: This element wraps the publications. It has no specified attributes.

publications: This element is a wrapper for a list of **category** instances being published. It has the following attribute:

uri (required): This attribute specifies the SIP URI of the publisher. This value MUST exactly match the To-URI and the From-URI.

publication: Each publication element describes a **category** instance to be published. It has the following attributes:

categoryName (required): This string value identifies the category to which the instance belongs.

instance (required): This unsigned integer specifies the instance number of the **category** instance. This value is used by the publisher to distinguish between distinct instances of the same **category** in a given container.

version (required): This unsigned integer specifies the version number of the **category** instance.

container (required): This unsigned short integer specifies the container to which the **category** instance is being published.

expireType (required): This string value determines how the lifetime of the **category** instance is bound. Valid values are "static", "endpoint", "user", and "time" and have the same semantics as in section 2.2.2.1.

expires (optional): This string value specifies the time, in Coordinated Universal Time (UTC) as prescribed by [ISO-8601], when this **category** instance will expire. This attribute MUST be specified for **category** instances where **expireType** has the value "time". If set to zero, this attribute indicates that the **category** instance is being unpublished, or removed. A nonzero value MUST NOT be specified unless the **expireType** attribute is set to "time".

The body of the **publication** element includes the data to be published. This data SHOULD be interpreted according to the schema identified by the **categoryName** attribute.

2.2.2.2.2 Response to a publish Request

For an example of a server 200 OK response, see section 4.2.1.

The **Content-Type** header field value MUST be "application/vnd-microsoft-roaming-self+xml", which means that this 200 OK carries an XML body that contains a roaming-self response that is fully specified in section 2.2.2.3.2.

Elements and attributes of a response to a category publish request are defined as follows.

The XML elements of a publish document response MUST conform to the schema defined in section 6.7.

roamingData: This element wraps the notification. It has no specified attributes.

categories: This element follows the **category** document schema described in section 2.2.2.1 and specifies the list of **category** instances being notified.

2.2.2.2.3 Version Conflict Response

A 409 Conflict response is a server response that indicates that a **category** instance version in a publication request is not equal to the server version of that **category** instance. The body of a 409 Conflict response contains information about the version number specified in the request and the current version number at the server. Section 4.2.1 provides an illustration of a 409 Conflict response message.

The **ms-diagnostics header** field is documented in [MS-OCER] section 2.2.1.1, and provides information that can be useful in further diagnostics. The **Content-Type** header field value MUST be "application/msrtc-fault+xml", which means that this 409 Conflict carries an XML body that contains an **msrtc-fault** response.

Elements and attributes of a 409 Conflict response are defined as follows.

Fault: This element wraps the fault information.

Faultcode: This element specifies the reason for the fault. A value of "Protocol client.BadCall.WrongDelta" is used to indicate the version number in the publish request is not equal to the current version number at the server.

details: This element is a wrapper for a list of faulting operations.

operation: Each operation element describes an operation that resulted in the fault. It has the following attributes:

index (required): This unsigned integer identifies the **category** instance in the triggering **publish** request.

version (required): This unsigned integer returns the version number specified for the **category** instance in the **publish** request.

curVersion (required): This unsigned integer specifies the current version number of the **category** instance maintained at the server. For a **publish** request to be successful, it SHOULD be retried with this value.

The body of the **operation** element includes current data for the **category** instance maintained at the server.

2.2.2.3 Self Subscription Related Messages

The following sections specify self subscription-related messages for enhanced presence.

2.2.2.3.1 Self Subscribe Request

The following sample shows bob@contoso.com subscribing to the **vnd-microsoft-roaming-self** event.

```
SUBSCRIBE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt -AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: vnd-microsoft-roaming-self
Accept: application/vnd-microsoft-roaming-self+xml
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization:...
Content-Type: application/vnd-microsoft-roaming-self+xml
Content-Length: …

<roamingList xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self">
  <roaming type="categories"/>
  <roaming type="containers"/>
  <roaming type="subscribers"/>
  <roamingEx xmlns="http://schemas.microsoft.com/2007/09/sip/roaming-self-ex"
type="delegates"/>
</roamingList>
```

Self subscription uses the SUBSCRIBE request, response and NOTIFY as defined in [RFC3265] section 3.1 and section 3.2, with the "vnd-microsoft-roaming-self" subscription **event package**.

The **Supported** and **Proxy-Require** header fields indicate support for a variety of presence extensions that are described in [MS-SIP] section 3.2.

The **Accept** header field MUST be set to "application/vnd-microsoft-roaming-self+xml" to indicate that the SIP protocol client is capable of receiving (in response to SUBSCRIBE) a roaming-self response (application/vnd-microsoft-roaming-self+xml).

The **Event** header field MUST be set to "vnd-microsoft-roaming-self" to indicate the self subscription event.

The **Content-Type** header field indicates that this SUBSCRIBE carries an XML body that contains user data for self use. It MUST be set to "application/vnd-microsoft-roaming-self+xml".

The To-URI and the From-URI carry the SIP URI of the watcher. The **From** and **To** header fields MUST be identical for the vnd-microsoft-roaming-self subscription event.

The SIP protocol client MUST add a valid **Contact** header field that can be used as a remote target URI of the SIP dialog route set as specified in [RFC3261] section 12.

Elements and attributes used in the XML body of the preceding sample self-SUBSCRIBE request are defined as follows. For the full XML schema, see section 6.

The XML elements of a self-subscribe request document MUST conform to the schema defined in section 6.6.

roamingList: This element is a wrapper for multiple **roaming** or **roamingEx** elements.

roaming: This element has a mandatory **type** attribute that indicates the type of roaming data that the subscriber is interested in. The value of **type** MUST be "categories", "containers", or "subscribers".

roamingEx: This element has a mandatory **type** attribute that indicates the type of roaming data that the subscriber is interested in. The value of **type** SHOULD be "delegates".

2.2.2.3.2 Self Subscribe Response

The following example shows the piggyback response that the server generates in response to a self SUBSCRIBE request:

```
SIP/2.0 200 OK
Contact:...
Authentication-Info: ...
Content-Length: …
From: "john"<sip:john@contoso.com>;tag=02025103a9;epid=c05eb044ab
To: <sip:john@contoso.com>;tag=F57F0080
Call-ID: 9f94d0c4f606472bb98a17bc07d6f96a
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TCP 10.88.20.31:3807;ms-received-port=3807;ms-received-cid=1500
Expires: 45792
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=45792
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:john@contoso.com">
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:08.217"
container="0" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>john</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="userProperties" instance="0" publishTime="2008-01-11T17:06:08.217"
container="1" version="1" expireType="static">
      <userProperties>
```

```
        <telephonyMode>None</telephonyMode>
      </userProperties>
    </category>
  </categories>
  <containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
    <container id="32000" version="0"/>
    <container id="400" version="0"/>
    <container id="300" version="0"/>
    <container id="200" version="0"/>
    <container id="100" version="0"/>
    <container id="1" version="0"/>
    <container id="0" version="0">
      <member type="everyone"/>
    </container>
  </containers>
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
    <subscriber user="bob@contoso.com" displayName="Bob" acknowledged="false"
type="sameEnterprise"/>
  </subscribers>
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="2">
    <delegate uri="bob@contoso.com" publish="false" redelegate="false"
  </delegates>
</roamingData>
```

The server MUST set the **Content-Type** header field value to "application/vnd-microsoft-roaming-self+xml". The XML body MUST be a valid http://schemas.microsoft.com/2006/09/sip/roaming-self document.

The XML body (application/vnd-microsoft-roaming-self+xml) consists of four subsections. They are **categories**, **containers**, **subscribers** and **delegates**. The schema definition for these elements is as follows:

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <xs:import
        namespace="http://schemas.microsoft.com/2006/09/sip/categories"
        schemaLocation="CategoriesNotification.xsd"/>
  <xs:import
        namespace="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
        schemaLocation="SubscribersNotification.xsd"/>
  <xs:import
        namespace="http://schemas.microsoft.com/2006/09/sip/containers"
        schemaLocation="ContainersNotification.xsd"/>
  <xs:import namespace="http://schemas.microsoft.com/2007/09/sip/delegates"
schemaLocation="DelegatesNotification.xsd"/>
  <xs:complexType name="RoamingDataType">
    <xs:sequence>
      <xs:element ref="cat:categories" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="con:containers" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="sub:subscribers" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="del:delegates" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="roamingData" type="RoamingDataType"/>
</xs:schema>
```

As this schema definition shows, the XML body subsections are XML documents from four different namespaces. For the details of these namespaces, see section 6.

The XML elements of a self-subscribe response document MUST conform to the schema defined in section 6.7.

Section 2.2.2.1 specifies elements and attributes of categories in the http://schemas.microsoft.com/2006/09/sip/categories namespace. Section 2.2.2.6.1 specifies elements and attributes of subscribers in the http://schemas.microsoft.com/2006/09/sip/presence-subscribers namespace. Section 2.2.2.5.1 specifies elements and attributes of containers in the http://schemas.microsoft.com/2006/09/sip/containers namespace. Section 2.2.2.10.1 specifies elements and attributes of delegates in the http://schemas.microsoft.com/2007/09/sip/delegates namespace. The application/vnd-microsoft-roaming-self+xml XML body is a wrapper around the XML subsections from these namespaces.

2.2.2.3.3 Self Subscribe Notify

When the server generates a NOTIFY or BENOTIFY request, it MUST set the **Content-Type** header field value to "application/vnd-microsoft-roaming-self+xml". The XML body MUST be a valid http://schemas.microsoft.com/2006/09/sip/roaming-self document.

[MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

The following example shows a BENOTIFY request generated as a result of a **note** category publication:

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK45F0BC36.F559A04F;branched=FALSE
Authentication-Info:...
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 15 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46487

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="200" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="300" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="400" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
```

```
  </categories>
</roamingData>
```

In the preceding example, the change that triggered the notification is a **category** publication, so the **roamingData** element has only a **categories** subelement. If notification is generated as a result of a container membership change, the body MUST have a containers XML document, as shown in the next example. Similarly, if notification is generated as a result of a subscriber list management operation, the body MUST have a **subscribers** XML document. Also if notification is generated as a result of a delegate management operation, the body MUST have a **delegates** XML document.

The following example shows a BENOTIFY request generated as a result of a container membership change:

```
BENOTIFY sip:172.24.32.124:54059;transport=tcp;ms-opaque=3a7258e2aa;ms-received-cid=D00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bKAA6EB91B.616A251C;branched=FALSE
Authentication-Info: ...
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=ea035dd987;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=0C390080
Call-ID: fb6533aac0034dd2a912d05d3f238a25
CSeq: 14 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=51706

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
    <container id="100" version="5">
      <member type="federated"/>
    </container>
    <container id="200" version="4">
      <member type="sameEnterprise"/>
      <member type="publicCloud"/>
    </container>
  </containers>
</roamingData>
```

2.2.2.4 Category Subscription-Related Messages

The following sections specify category subscription-related messages for enhanced presence.

2.2.2.4.1 Category Subscribe Request

The following example shows a category subscription request.

```
SUBSCRIBE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:54059
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=26aef0a2a4;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 77a9ac0f3beb4a65b1d40cc0978eda56
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt -AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/rlmi+xml, multipart/related
```

```
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: ...
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:bob@contoso.com" name="">
  <action name="subscribe" id="63792024">
    <adhocList>
      <resource uri="sip:alice@contoso.com"/>
      <resource uri="sip:john@contoso.com">
        <context>
     <subscriptionContext
xmlns="http://schemas.microsoft.com/2008/09/sip/SubscriptionContext" majorVersion="1"
minorVersion="0">
            <watcher>
              <contactList />
            </watcher>
     </subscriptionContext>
   </context>
      </resource>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

Category subscription uses the SUBSCRIBE request and response as defined in [RFC3265] section 3.1 and section 3.2, with the presence event package. The value of the **Event** header field MUST be "presence".

The **Supported**, **Require**, and **Proxy-Require** header fields indicate support for a variety of presence extensions that are described in [MS-SIP] section 3.2.

The **Accept** header field indicates that the SIP protocol client is capable of receiving, in response to SUBSCRIBE, a multipart **MIME** document that contains a list of users ("application/rlmi+xml") and their **category** data ("application/msrtc-event-categories+xml"). An **Accept** header field MUST include "application/msrtc-event-categories+xml, application/rlmi+xml, multipart/related".

The **Require** header field MUST be set to "adhoclist, categoryList".

The **Supported** header field MUST be set to "eventlist". This indicates support specifically for the enhanced presence batched SUBSCRIBE mechanism.

The **Content-Type** header field value of "application/msrtc-adrl-categorylist+xml" indicates that this SUBSCRIBE request carries an XML body that contains the list of contacts of interest, along with categories. **Content-Type** MUST be set to "application/msrtc-adrl-categorylist+xml".

The **To-URI** and the **From-URI** MUST contain the SIP URI of the subscriber. In the previous example, bob@contoso.com is the subscriber.

The SIP protocol client MUST add a valid **Contact** header field that can be used as a remote target URI of the SIP dialog route set as specified in [RFC3261] section 12.

Elements and attributes used in category SUBSCRIBE requests are defined as follows. For the full **XML schema**, see section 6.

The XML elements of a category SUBSCRIBE request document MUST conform to the schema defined in section 6.8.

batchSub: This element is a wrapper. It has two optional attributes. If present, these attributes MUST be ignored by the server.

- **uri:** Subscriber's SIP URI
- **name:** Subscriber's name

action: This element defines the subscription action. It has two attributes:

- **name:** MUST be "subscribe" or "unsubscribe".
- **id:** This attribute is ignored by the server if present in the request.

adhocList: This is the wrapper for the resource list.

resource: This element has a URI attribute that is the SIP URI to be subscribed to or unsubscribed from. It is possible to have more than one of these listed, as shown in the preceding example.

context: This is a child-level XML element of a **resource** element. If the subscriber intends to be added to the resource's subscriber list, the subscriber SHOULD add this element. When the context element is included, it SHOULD contain a **subscriptionContext** element with a **watcher** element, which indicates that a contact has been added as a watcher.<2> If the contact has not previously subscribed to the list, this element triggers the server to notify the owner of the list that the contact is now a subscriber.

If the context element is empty, an implicit assumption is made that the contact specified by **uri** attribute in **batchSub** element has previously successfully subscribed to the corresponding **resource**.

In the preceding example, bob@contoso.com is requesting to be added to the subscriber list of john@contoso.com. Subscriber lists are specified in section 2.2.2.6 and section 3.6.

When the resource is a delegate of the subscriber, the context SHOULD additionally contain a **delegation** element to notify that the delegate relationship has been set up. This is the primary way to notify a delegate about the permission setting. The delegation node MUST be empty.

```
<context>
     <subscriptionContext
xmlns="http://schemas.microsoft.com/2008/09/sip/SubscriptionContext" majorVersion="1"
minorVersion="0">
            <watcher>
              <contactList />
            </watcher>
            <delegation />
     </subscriptionContext>
</context>
```

More information about delegation is covered in the [MS-SIPRE] section 3.9.5.2.2.2. The **SubscriptionContext** schema is described in Section 6.13.

categoryList: This element is a wrapper for the list of **category** elements.

category: This element represents the category to be subscribed to. It has one required attribute, **name**, that holds the name of the **category**. There can be multiple **category** elements, as shown in the previous example.

2.2.2.4.2 Category Subscribe Response

The server MUST set the **Content-Type** header field to "multipart/related; type="application/rlmi+xml";start=resourceList;". The server MUST also append the MIME boundary to the **Content-Type** header field as shown in the following example and MUST use the same separator between documents in the body. For details about MIME, see [RFC1341] Appendix E.

The following is an example for a category subscription response.

```
SIP/2.0 200 OK
Contact: ...
Authentication-Info: ...
Content-Length: …
From: "Bob"<sip:bob@contoso.com>;tag=26aef0a2a4;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=99000080
Call-ID: 77a9ac0f3beb4a65b1d40cc0978eda56
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TCP 172.24.32.124:54059;ms-received-port=54059;ms-received-cid=D00
Expires: 21888
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=fd6a7790014f493faeaf225c3a2eb3d9
Event: presence
subscription-state: active;expires=21888
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

--fd6a7790014f493faeaf225c3a2eb3d9
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml
<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:bob@contoso.com" version="0"
fullState="false">
  <resource uri="sip:john@contoso.com">
    <instance id="0" state="resubscribe" cid="john@contoso.com
poolFqdn="sip:server3.contoso.com@contoso.com;gruu;opaque=srvr:HomeServer:dL8cwxBrTuG8eC4
-Q_GNGAAA"/>" />
  </resource>
</list>

--fd6a7790014f493faeaf225c3a2eb3d9
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml
<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:alice@contoso.com">
  <category name="calendarData"/>
  <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>Alice</displayName>
        </name>
      </identity>
    </contactCard>
  </category>
  <category name="note"/>
  <category name="state" instance="0" publishTime="2008-01-11T17:10:49.560">
    <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>18000</availability>
    </state>
  </category>
  <category name="services"/>
</categories>
--fd6a7790014f493faeaf225c3a2eb3d9--
```

A response to a batched subscription has a multipart MIME body that contains both a list of resources and the category data for each of those contacts.

The first part has a list of resources for rejected subscriptions. Elements and attributes are defined later in this section. For the full XML schema, see section 6.

The XML elements of a category subscribe response document MUST conform to the schema defined in section 6.9.

list: This element is a wrapper with the following attributes:

- **uri:** The SIP URI of the subscriber.
- **version:** A nonnegative integer.
- **fullState:** This attribute MUST be set to **false**.
- **cid (optional):** Ignored.

resource: This element represents a resource from the request body. Resource elements exist only if one or more of the subscriptions listed in the batch category SUBSCRIBE request is rejected. It has one attribute:

- **uri:** Resource URI.

instance: One instance element is present for each resource. It has these attributes:

- **id:** This attribute MUST be set to 0.
- **state:** Contains the subscription **state** for the identified instance of the resource. It MUST contain one of these values: "active", "pending", "terminated", "resubscribe".
- **reason:** If the **state** attribute is set to "terminated", then a reason attribute MUST also be present. This is for informational purposes only. SIP protocol clients are not expected to take any automated action based on its value.
- **statusCode:** Provides the status code (if available) for the reason that a subscription to a particular resource was not accepted by the server. For example, "413" can be returned when a resource exceeds an incoming subscription limit. Note that this behavior is implementation dependent.
- **cid:** The resource URI.
- **poolFqdn:** This attribute MUST supply the URI that routes to a server (or server pool) that can accept a batch subscription for the user. The server uses a pool **GRUU** for this purpose. This attribute is needed only if the resource is not collocated with the publisher on the same server or server pool. SIP protocol clients MAY use this information to send a batched category SUBSCRIBE request to the provided URI. When **poolFqdn** is specified, the value of the **state** attribute MUST be "resubscribe".

The rest of the body is a delimited list of **category** data. Each delimited part MUST have its **Content-Type** header field set to "application/msrtc-event-categories+xml". The server MUST put the **categories** document associated with one user in each delimited part. **Content-Transfer-Encoding** MUST be set to "binary". For the definition of the "application/msrtc-event-categories+xml" document format, refer to section 2.2.2.1. The server MUST return **category** data for all resources specified in the request except those that are in **list**, which contains those resources that are rejected.

2.2.2.4.3 Category Subscribe Notify

The server (2) MUST set the **Content-Type** header field value to "application/msrtc-event-categories+xml". The XML body MUST be a valid

http://schemas.microsoft.com/2006/09/sip/categories document. A detailed description of this **content type** header can be found in section 2.2.2.1.

[MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

Here is a sample BENOTIFY sent to bob@contoso.com (**category subscriber**) in response to a publication by user2@contoso.com:

```
BENOTIFY sip:172.24.32.124:54111;transport=tcp;ms-opaque=d445641ebd;ms-received-cid=1100
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK1A8DAF2D.8C27980A;branched=FALSE
Authentication-Info: ...
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=2e359b464c;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=1F700080
Call-ID: cccb942d2c044a8a8edb6828f7496d06
CSeq: 6 BENOTIFY
Require: eventlist
Content-Type: application/msrtc-event-categories+xml
Event: presence
subscription-state: active;expires=31927

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:user2@contoso.com">
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
            name="state" instance="1"
            publishTime="2008-01-11T18:11:33.577">
    <state xsi:type="aggregateState"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>6500</availability>
      <activity token="in-a-meeting"/>
      <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
      <meetingSubject>Customer Meeting</meetingSubject>
      <meetingLocation>Conference Room - Building 7</meetingLocation>
    </state>
  </category>
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
            name="services" instance="0"
            publishTime="2008-01-11T18:04:53.780">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:user2@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500"/>
          <gifInk render="true" capture="false" deviceAvailability="3500"/>
          <isfInk render="true" capture="false" deviceAvailability="3500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>
```

2.2.2.5 Container Management-Related Messages

The following sections specify container management-related messages for enhanced presence.

2.2.2.5.1 containers Document

A number of enhanced presence SIP messages indicate a list of containers. The following example illustrates the schema used for this purpose.

```
<containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
  <container id="100" version="1">
```

```
    <member type="federated"/>
  </container>
  <container id="200" version="1">
    <member type="sameEnterprise"/>
  </container>
</containers>
```

Elements and attributes of a **containers** document are defined in this section.

The XML elements of a **containers** document MUST conform to the schema defined in section 6.3

containers: This element is a wrapper for a list of containers. Multiple instances of the **containers** element MAY exist in a single request. This element has no specified attributes.

container: Each **container** element describes a container. It wraps a list of members that belong to the **container**. It also has the following attributes:

- **id** (required)**:** This unsigned short integer uniquely identifies a **container**.
- **version** (required): This unsigned integer specifies the version number of the **container** maintained by the server. The version number allows multiple endpoints of the same publisher to synchronize their publications and allows self subscribers to determine whether to accept a notification.

member: Each member element describes a **container** member and has the following attributes:

- **type** (required): This string value specifies the type of the member. The following values are **defined** and MUST be supported by the server following the description given here:
  - **user:** A specific SIP URI.
  - **domain:** A specific SIP domain.
  - **sameEnterprise:** All users from the same enterprise.
  - **federated:** All federated users.
  - **publicCloud:** All public cloud users.
  - **everyone:** All users.
- **value** (optional): This string value further qualifies the **member** element. A **member** element of **type** "domain" MUST have a value that specifies a domain suffix. A **member** element of **type** "user" MUST have a value that specifies a SIP URI. The value MUST be omitted for all other **type** values.

2.2.2.5.2 setContainerMember Request

A publisher uses a SERVICE request to manage container membership. The following example illustrates the format of such a message:

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=7e4e527abf;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 4c4fc82adeef42c9b3859f080c74a68c
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu="">
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
```

```
Content-Type: application/msrtc-setcontainermembers+xml
Content-Length: …

<setContainerMembers xmlns="http://schemas.microsoft.com/2006/09/sip/container-
management">
  <container id="300" version="0">
    <member action="add" type="user" value="alice@contoso.com"/>
  </container>
</setContainerMembers>
```

The **To** and **From** header fields MUST carry the same SIP URI, which MUST be that of the publisher. The **Content-Type header** MUST have a value of "application/msrtc-setcontainermembers+xml", which means that this SERVICE carries an XML body that contains a list of actions for managing container membership.

Elements and attributes in the XML body are defined in this section. For the full XML schema, see section 6.

The XML elements of a set container member document MUST conform to the schema defined in section 6.11.

setContainerMembers: This element is a wrapper for a list of container membership actions. It has no specified attributes.

container: This element is a wrapper for a list of actions to be performed to a container membership. It has the following attributes:

- **id** (required): This unsigned short integer uniquely identifies the container whose membership is being modified. The membership of the default container, which is container zero, cannot be modified, and hence **id** MUST NOT be zero.
- **version** (required): This unsigned integer specifies the version number of the container membership.

member: Each member element describes an action that alters the container membership. It has the following attributes:

- **type (required):** This string value specifies the type of the member. Valid values are "user", "domain", "sameEnterprise", "federated", "publicCloud", and "everyone" and have the same semantics as described in section 2.2.2.5.1.
- **value (optional):** This string value further qualifies the member and has the same semantics as in section 2.2.2.5.1.
- **action (optional):** This string value specifies the action to be performed with the member. The following values are defined and MUST be supported by the server following the description given here:
  - **add:** Request that the specified member be added to the container membership. The server treats this as the default action.
  - **delete:** Request that the specified member be deleted from the container membership.

If the operation is successful, the server MUST return a 200 OK response.

2.2.2.5.3 Version Conflict Response

A 409 Conflict response is a server response that indicates that version number specified for the container membership in setContainerMember request is not equal to the server version. The body of a 409 Conflict response contains information about the version number specified in the request and

the current version number at the server. For an example of a 409 Conflict response message, see section 4.2.1.

The ms-diagnostics header is documented in [MS-OCER] section 2.2.1.1, and provides information that can be useful in further diagnostics. The **Content-Type** header field MUST have a value of "application/msrtc-fault+xml", which means that this 409 Conflict carries an XML body that contains an **msrtc-fault** response.

Elements and attributes of a 409 Conflict response are defined as follows.

Fault: This element wraps information about the fault.

Faultcode: This element specifies the reason for the fault. A value of "Protocol client.BadCall.WrongDelta" is used to indicate the version number in the publish request is not equal to the current version number at the server.

details: This element is a wrapper for a list of faulting operations.

operation: Each **operation** element describes an operation that resulted in the fault. This element has the following attributes:

- **index (required):** This unsigned integer identifies the list of container membership actions in the triggering **setContainerMember** request.
- **version (required):** This unsigned integer returns the version number specified for the container membership in the **setContainerMember** request.
- **curVersion (required):** This unsigned integer specifies the current version number of the container membership maintained at the server. For a **setContainerMember** request to be successful, it SHOULD be retried with this value.

2.2.2.6 Subscriber List

The following sections specify subscriber list management-related messages for enhanced presence.

2.2.2.6.1 subscribers Document

The **subscribers** document XML schema is as follows:

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="ContextType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="1"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="SubscriberType">
    <xs:sequence>
      <xs:element name="context" minOccurs="0" maxOccurs="1"
type="ContextType"></xs:element>
    </xs:sequence>
    <xs:attribute name="user" type="xs:string" use="required"></xs:attribute>
    <xs:attribute name="displayName" type="xs:string" use="required"></xs:attribute>
    <xs:attribute name="acknowledged" type="xs:boolean" use="required"></xs:attribute>
    <xs:attribute name="type" use="optional">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="federated"></xs:enumeration>
          <xs:enumeration value="publicCloud"></xs:enumeration>
```

```
            <xs:enumeration value="sameEnterprise"></xs:enumeration>
            <xs:enumeration value="unknown"></xs:enumeration>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
    <xs:complexType name="SubscribersType">
      <xs:sequence>
        <xs:element name="subscriber" type="SubscriberType" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      </xs:sequence>
    </xs:complexType>
    <xs:element name="subscribers" type="SubscribersType"></xs:element>
</xs:schema>
```

The following is an example **subscribers** document:

```
<subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscriber  user="bob@contoso.com"
               displayName="Bob"
               acknowledged="false"
               type="sameEnterprise"/>
</subscribers>
```

Elements and attributes of a **subscribers** document are defined in this section.

The XML elements of a **subscribers** document MUST conform to the schema defined in section 6.4

subscribers: This element is a wrapper for multiple **subscriber** elements.

subscriber: This element provides information about a subscriber. The attributes are the following:

- **user:** The URI of the **subscriber**. This value MUST be a valid URI.
- **displayName:** The display name of the **subscriber**. The server can use the display name received in the SUBSCRIBE request.
- **acknowledged:** A Boolean value that shows whether the **subscriber** has been acknowledged by the publisher. A value of **true** denotes that the **subscriber** has been acknowledged by the publisher; **false** denotes that the subscriber has not been acknowledged.
- **type:** This is an optional attribute. If the server can determine the source network for the incoming SUBSCRIBE request, it MAY provide this additional information. If supplied, the value is one of the following: "federated", "publicCloud", "sameEnterprise" and "unknown".
- **context:** A copy of a context element in a SUBSCRIBE request from this subscriber.

2.2.2.6.2 setSubscribers Request

SIP SERVICE request and response messages are used to manage subscriber lists. In the following example, Bob acknowledges john@contoso.com.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:54111
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=aa5b2cc9ef;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 345dfe18894e4e608ae845de4f6108fa
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
```

```
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-presence-setsubscriber+xml
Content-Length: …

<setSubscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscriber user="john@contoso.com" acknowledged="true"/>
</setSubscribers>
```

The **Content-Type** header field MUST be set to "application/msrtc-presence-setsubscriber+xml ". The XML body MUST be a valid http://schemas.microsoft.com/2006/09/sip/presence-subscribers document.

The **To-URI** and the **From-URI** MUST contain the SIP URI of the user who acknowledges the subscriber.

A **setSubscribers** request does not support batching, or acknowledging multiple subscribers at the same time.

Key elements and attributes are defined in this section. For the full XML schema, see section 6.

The XML elements of a set subscribers document MUST conform to the schema defined in section 6.12.

subscriber: This element is used to describe the operation. It has two attributes:

- **user:** This mandatory attribute in the subscriber element is the URI for the subscriber to be acknowledged. The value MUST be a valid URI.
- **acknowledged:** The value of this attribute MUST be **true**.

If the operation is successful, the server MUST return a 200 OK response.

2.2.2.7 Categories

Presence information of a user is organized into various categories. The server is normally agnostic of the XML schemas for SIP protocol client publications. There are exceptions to this. The server has the knowledge of the XML schemas for the categories listed in the following sections. Some of these categories, such as the **state** and **device** categories, are required for aggregation,some categories, such as **userProperties,** are published by the server, and some categories, such as **legacyInterop,** are needed to generate the XML body for **MSRTC/PIDF** notifications. The following sections contain details for each of these category types.

The server MUST allow categories to be marked as "private".

2.2.2.7.1 state

The **state** category is used to represent the various presence states associated with a user. The server aggregates instances of the **state** category to compute a user's overall presence state. For details about aggregation, see section 3.8.5.

The following examples illustrate the format of a **state** category instance.

```
<state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
       manual="false"
       uri="bob@contoso.com"
       startTime="2008-01-11T19:00:00Z"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:type="calendarState">
  <availability>6500</availability>
```

```
  <activity token="in-a-meeting" minAvailability="6500" maxAvailability="8999" />
  <endpointLocation>
  </endpointLocation>
  <meetingSubject>Customer Meeting</meetingSubject>
  <meetingLocation>Conf Room 100</meetingLocation>
</state>

<state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       manual="false"
       xsi:type="machineState">
  <availability>5000</availability>
  <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
  <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  <timeZoneBias>999</timeZoneBias>
          <timeZoneName>Pacific Daylight Time</timeZoneName>
          <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
          <device>computer</device>
  <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/></state>

<state xsi:type="aggregateState"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://schemas.microsoft.com/2006/09/sip/state">
  <availability>9500</availability>
  <activity>
    <custom LCID="1033">Interviewing</custom>
  </activity>
  <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  <timeZoneBias>999</timeZoneBias>
          <timeZoneName>Pacific Daylight Time</timeZoneName>
          <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
          <device>computer</device>
  <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
</state>

<state xsi:type="aggregateMachineState"
       endpointId="221ef77e-3a68-5570-86ed-6ea5bd4b7ff8"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://schemas.microsoft.com/2006/09/sip/state">
  <availability>5000</availability>
</state>
```

Elements and attributes are defined as follows. For the full XML schema, see section 7.

The XML elements of a **state** document MUST conform to the schema defined in section 7.2

state: The root element of a **state** category instance. It has the following attributes:

- **xsi:type (required):** The type of the **state** element. It MUST have one of the following values:
  - **userState:** This instance type reflects a presence state that was manually set by the user.
  - **machineState:** This instance type reflects the presence state of the physical device used by a user. It indicates how active an endpoint is at the current time. The SIP protocol client monitors keyboard and mouse activity to determine the value of this state. The SIP protocol client publishes this type of instance for each of a user's endpoints.
  - **calendarState:** This instance type reflects the presence state of user's calendar. The SIP protocol client uses the user's calendar information to publish this type of instance.

- **phoneState:** This instance type reflects the presence state of the user's voice or video conversations. The SIP protocol client publishes this type of instance based on the voice or video conversations in which the user is currently engaged.
- **aggregateState:** This instance type represents the user's overall presence state.
- **aggregateMachineState:** This instance type represents the overall presence state of all physical devices used by a user. This presence state is an aggregation(2) of all of the **machineState** instance types.
- **presentingState:** This instance type of presence state indicates that user is in a presentation.<3>

- **uri (optional):** The URI associated with the **phoneState** and **calendarState** instances. It specifies the mailbox identifier in **calendarState** instance types and the **tel** URI in **phoneState** instance types.
- **manual (optional):** Specifies whether the state is published by a user manually, **true** or **false**.
- **startTime (optional):** Specifies the starting time when the state became effective.
- **endpointId (optional):** Specific to **aggregateMachineState** instance type. This is an endpoint identifier of the most active **machineState** chosen by the computing function when creating **aggregateMachineState** as described in section 3.8.5.
- **lastActive (optional):** The time, in UTC, that the user became unavailable for communications. This attribute is specific to the **aggregateState** instance type.
- **majorVersion (optional):** The major version of schema format.
- **minorVersion (optional):** The minor version of schema format.
- **availability:** A child element of the **state** element. It contains a number representing the **availability** of a presence state as follows:
  - **Unknown:** 0 to 2999. **availability** is undefined.
  - **Online:** 3000 to 4499. Willing and able to communicate.
  - **Idle:** 4500 to 5999. Willing but potentially unable to communicate.
  - **Busy:** 6000 to 7499. Able but potentially unwilling to communicate.
  - **BusyIdle:** 7500 to 8999. Able but potentially unwilling to communicate.
  - **DoNotDisturb:** 9000 to 11999. Able but potentially unwilling to communicate.
  - **Away:** 12000 to 17999. Willing but unable to communicate.
  - **Offline:** 18000 and higher. Not available to communicate.
- **activity:** A **child element** of the **state** element. It contains the activities that triggered the state. It has the following attributes:
  - **token (optional):** Localizable **token** that represents the activity string of a **state**.
  - **maxAvailability (optional):** Highest **availability** for which this activity is valid. For details about how this attribute is used, see section 3.8.5.
  - **minAvailability (optional):** Lowest **availability** for which this activity is valid. For details about how this attribute is used, see section 3.8.5.

- **custom:** A child element of the **activity** element. It represents a language-specific string which is associated with an LCID when **token** is absent. It has the following attributes:
  - **LCID (optional):** Locale identifier for the activity string.
  - **updated (optional):** Last time, specified in UTC, that the custom element was updated.
- **endpointLocation:** A child element of the **state** element. This represents the location of a device associated with the user's endpoint. The **endpointLocation** element SHOULD be published by the SIP protocol client as part of its machine state. This element is specific to the **machineState** and **aggregateState** instance types.
- **meetingSubject:** A child element of the **state** element. This represents the subject of the user's calendar meeting. This element is specific to the **calendarState** and **aggregateState** instance types. It MAY have the following attributes.
  - **LCID (optional):** Locale identifier for the meeting subject.
  - **updated (optional):** Last time, specified in UTC, that the **meetingSubject** element was updated.
- **meetingLocation:** A child element of the **state** element. This represents the location of the calendar meeting. This element is specific to **calendarState** and **aggregateState** instance types. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier for the meeting location.
  - **updated (optional):** Last time, specified in UTC, that the **meetingLocation** element was updated.
- **timeZoneBias:** A child element of the **state** element. The value of this element is the difference in minutes between UTC and local time, either standard time or daylight saving time, of the publisher.
- **timeZoneName:** A child element of the **state** element. The value of this element contains the name or description for the time zone, either standard time or daylight saving time, of the publisher.
- **timeZoneAbbreviation:** A child element of the **state** element. The value of this element contains the abbreviation for the time zone, either standard time or daylight saving time, of the publisher.
- **device:** A child element of the **state** element. This identifies the type of the endpoint device. It SHOULD<4> have one of the following values:
  - **computer:** Endpoint is the computer desktop.
  - **deskphone:** Endpoint is the phone device.
  - **mobile:** Endpoint is the mobile device.
  - **web:** Endpoint is the web browser.

- **treatLocationAsProximate:** A child element of the **state** element. This element is specific to **machineState.** The value of this element specifies whether location information in this **machineState** category instance is treated as proximately and is used only when user is not available in any other endpoints.

2.2.2.7.2 device

The **device** category contains information about the capability of the physical device used by a user. The server aggregates the instances of the **device** category to produce the **services** category, as specified in section 3.8.5.2.

The following examples illustrate the format of a **device** category instance.

```
<device xmlns="http://schemas.microsoft.com/2006/09/sip/device"
          endpointId="2CD4F7CA-B1D1-5EDA-8D79-79EE4298414D">
  <capabilities preferred="false" uri="sip:bob@contoso.com">
    <text capture="true" render="true" publish="false">
    </text>
    <gifInk capture="false" render="true" publish="false">
    </gifInk>
    <isfInk capture="false" render="true" publish="false">
    </isfInk>
    <breakthrough render="false" capture="false" publish="true">
    </breakthrough>
    <applicationSharing render="true" capture="true" publish="false">
    </applicationSharing>
  </capabilities>
  <capabilities preferred="false" uri="bob@exchange.contoso.com">
    <calendar capture="false" render="false" publish="true" version="655616" />
  </capabilities>
  <timezone>00:00:00-00:00</timezone>
  <machineName>BOB-B</machineName>
</device>

<device xmlns="http://schemas.microsoft.com/2006/09/sip/device"
          endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">

  <capabilities preferred="false" uri="sip:bob@contoso.com">
    <text capture="true" render="true" publish="false">
    </text>
    <gifInk capture="false" render="true" publish="false">
    </gifInk>
    <isfInk capture="false" render="true" publish="false">
    </isfInk>
    <breakthrough render="false" capture="false" publish="true">
    </breakthrough>
    <applicationSharing render="true" capture="true" publish="false">
    </applicationSharing>
  </capabilities>
  <capabilities preferred="false" uri="tel:+11234567890;ext=67890">
    <remoteCallControl capture="false" render="false" publish="true">
    </remoteCallControl>
  </capabilities>
  <timezone>00:00:00-00:00</timezone>
  <machineName>BOB-A</machineName>
</device>
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a device document MUST conform to the schema defined in section 7.4

- **device:** This is a root element of **device** category data. It has the following attribute:
  - **endpointId (required):** The endpoint identifier of the device.
- **capabilities:** This element lists the device's capabilities. The schema for the **capabilities** element in the **device** category is identical to the schema for the **capabilities** element in the **services** category. For details about the **services** category, see section 2.2.2.7.3.
- **timezone:** This element SHOULD NOT be used. If it is used, it is set to the following format when sending and ignored on receipt.

00:00:00-*NN:NN:NN* or 00:00:00+*NN:NN:NN*

Where *N* represents a numeric digit that can range from zero (0) to 9. The first NN represents two digits of the hour of the time zone. The second NN represents two digits of the minute of the time zone. The third NN represents two digits of the second of the time zone.

- **machineName:** This element contains the name of the physical device used by a user to connect to the server.

2.2.2.7.3 services

The **services** category is used to specify presence capabilities. The **services** category has the list of individual **service** elements. Each **service** element MUST correspond to a URI. For details about how a **services** category instance is computed, see section 3.8.5.2.

The following examples illustrate the format of a **services** category instance.

```
<services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
  <service uri="sip:bob@contoso.com">
    <capabilities>
        <text render="true" capture="true" publish="false"
              preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
              deviceAvailability="3500" />
        <gifInk render="true" capture="false" publish="false"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                deviceAvailability="3500" />
        <isfInk render="true" capture="false" publish="false"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                deviceAvailability="3500" />
        <breakthrough render="false" capture="false" publish="true"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                deviceAvailability="3500" />
        <applicationSharing render="true" capture="true" publish="false"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                deviceAvailability="3500" />    </capabilities>
  </service>
  <service uri="tel:+11234567890;ext=67890">
    <capabilities>
       <remoteCallControl render="false" capture="false" publish="true"
                          preferredEndpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
                          deviceAvailability="3500" />
    </capabilities>
  </service>
  <service uri="bob@exchange.contoso.com">
    <capabilities>
      <calendar render="false" capture="false" publish="true"
                version="655616"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                deviceAvailability="3500" />
    </capabilities>
  </service>
</services>

<services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
  <service uri="sip:bob@contoso.com">
    <capabilities>
      <text render="true" capture="true" deviceAvailability="3500" />
      <gifInk render="true" capture="false" deviceAvailability="3500" />
      <isfInk render="true" capture="false"  deviceAvailability="3500" />
    </capabilities>
  </service>
</services>
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a **services** document MUST conform to the schema defined in section 7.5.

- **services:** This is the root element. This element contains the list of **service** elements with their **capabilities**. It has the following attributes:
  - **majorVersion (optional):** The major version of schema format.
  - **minorVersion (optional):** The minor version of schema format.
- **service:** It has the following attribute:
  - uri (required): URI of the service.

- **capabilities:** The list of capabilities, each of which SHOULD<5> be represented by a **text** element, **voice** element, **video** element, **calendar** element, **remoteCallControl** element, **CCCP** element, **gifInk** element, **isfInk** element, **breakthrough** element, **applicationSharing** element, **ucs** element, **containerIntegrity** element, **contentWhiteboard** element, **contentPowerPoint** element, **contentNativeFile** element, **contentPoll** element, or **contentSharedNotes** element within the capabilities element.
  - **text:** The instant messaging capability of an endpoint device. It has the following attributes:
    - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving instant messages (true) or not (false).
    - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sending instant messages, **true** or **false**.
    - **publish (optional):** Ignored.
    - **version (optional):**<6> Ignored.
    - **deviceAvailability (optional):** The availability of the device associated with this capability.
    - **preferredEndpointId (optional):** Identifier of preferred client endpoint for auto-accepting instance messages.
  - **voice:** The capability of an endpoint device to make voice communication. It contains the following attributes:
    - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving voice communication, **true** or **false**.
    - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sending voice communication, **true** or **false**.
    - **publish (optional):** Ignored.
    - **version (optional):** Ignored.
    - **deviceAvailability (optional):** The availability of the device associated with this capability.
    - **preferredEndpointId (optional):** Ignored.
  - **video:** The capability of an endpoint device to make video communication. It contains the following attributes:
    - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving video communication, **true** or **false**.

- **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sending video communication, **true** or **false**.
- **publish (optional):** Ignored.
- **version (optional):** Ignored.
- **deviceAvailability (optional):** The availability of the device associated with this capability.
- **preferredEndpointId (optional):** Ignored.

- **calendar:** The capability of an endpoint device to publish calendar information to presence. It contains the following attributes:
  - **render (optional):**Ignored.
  - **capture (optional):**<7> Ignored.
  - **publish (optional):** A Boolean value that indicates whether the endpoint device is capable of publishing calendar information to presence, **true** or **false**.
  - **version (optional):** The version of the capability. Server MUST prefer the endpoint with higher version as a preferred endpoint as specified in section 3.8.5.2.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Identifier of preferred client endpoint for publishing calendar information to presence.
- **remoteCallControl:** The remote call control of an endpoint device. It has the following attributes:
  - **render (optional):** Ignored.
  - **capture(optional):**<8> Ignored.
  - **publish (optional):** A Boolean value that indicates whether the endpoint device is capable of publishing remote call control information to presence, **true** or **false**.
  - **version (optional):** The version of the capability. Server MUST prefer the endpoint with higher version as a preferred endpoint as specified in section 3.8.5.2.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Identifier of preferred client endpoint for publishing remote call control information to presence.
- **CCCP:** Ignored.
- **gifInk, isfInk:** The ink capability, or handwriting capability on a device using a stylus, of an endpoint device.
  - **render (optional):** A Boolean value that indicates whether rendering of this ink capability is supported, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether capturing this ink capability is supported, **true** or **false**.
  - **publish (optional):** Ignored.

- **version (optional):**<9> Ignored.
- **deviceAvailability (optional):** The availability of the device associated with this capability.
- **preferredEndpointId (optional):** Ignored.

- **breakthrough:** The capability of an endpoint device to publish **breakthrough** list in the **routing** category instance. Refer to [MS-SIPRE] section 3.9.5.1.4 for details about the **breakthrough** list. It has the following attributes:
  - **render (optional):** Ignored.
  - **capture (optional):** Ignored.
  - **publish (optional):** A Boolean value that indicates whether the endpoint device is capable of publishing **breakthrough** list, **true** or **false**.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):** Ignored.
  - **preferredEndpointId (optional):** Identifier of preferred client endpoint for publishing **breakthrough** list.
- **applicationSharing:** The application sharing capability of an endpoint device. It has the following attributes:
  - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of rendering application sharing, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of controlling or initiating application sharing with other users, **true** or **false**.
  - **publish (optional):** Ignored.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):**<10> The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Ignored.
- **ucs:** The capability of an endpoint device to create contacts in the mail server for contact list entries in the server. It has the following attributes:
  - **render (optional):** Ignored.
  - **capture (optional):** Ignored.
  - **publish (optional):** A Boolean value that indicates whether the endpoint device is capable of creating contacts on the mail server for contact list entries.
  - **version (optional):** The version of the capability. Server MUST prefer the endpoint with higher version as a preferred endpoint as specified in section 3.8.5.2.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Identifier of preferred client endpoint for creating contacts on the mail server.

- **containerIntegrity:** The capability of an endpoint device to create container membership according to rules defined in the **publicationGrammar** provision group in [MS-SIPREGE] section 2.2.2.5.4, or **privacyPublicationGrammar** provision group in section [MS-SIPREGE] section 2.2.2.5.10. It has the following attributes:
  - **render (optional):** Ignored.
  - **capture (optional):** Ignored.
  - **publish (optional):** A Boolean value that indicates whether the endpoint device is capable of creating container membership according to rules defined in the publication grammar provision groups.
  - **version (optional):** The version of the capability. Server MUST prefer the endpoint with higher version as a preferred endpoint as specified in section 3.8.5.2.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Identifier of preferred client endpoint for creating container membership.
- **contentWhiteboard<11>:** The whiteboard content capability of an endpoint device. It has the following attributes:
  - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving whiteboard content, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sharing whiteboard content with other users, **true** or **false**.
  - **publish (optional):** Ignored.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Ignored.
- **contentPowerPoint:<12>** The PowerPoint content capability of an endpoint device. It has the following attributes:
  - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving PowerPoint content, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sharing PowerPoint content with other users, **true** or **false**.
  - **publish (optional):** Ignored.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Ignored.
- **contentNativeFile:<13>** The capability of an endpoint device to receive and share opaque binary content in a multiparty conference. It has the following attributes:

- **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving opaque binary content in a multiparty conference, **true** or **false**.
- **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sharing opaque binary content in a multiparty conference with other users, **true** or **false**.
- **publish (optional):** Ignored.
- **version (optional):** Ignored.
- **deviceAvailability (optional):** The availability of the device associated with this capability.
- **preferredEndpointId (optional):** Ignored.

- **contentPoll<14>:** The capability of an endpoint device to participate in interactive polls. It has the following attributes:
  - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving polling data, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sharing polling data with other users, **true** or **false**.
  - **publish (optional):** Ignored.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
- **preferredEndpointId (optional):** Ignored.
- **contentSharedNotes<15>:** The shared notes content capability of an endpoint device. It has the following attributes:
  - **render (optional):** A Boolean value that indicates whether the endpoint device is capable of receiving shared notes content, **true** or **false**.
  - **capture (optional):** A Boolean value that indicates whether the endpoint device is capable of sharing shared notes content with other users, **true** or **false**.
  - **publish (optional):** Ignored.
  - **version (optional):** Ignored.
  - **deviceAvailability (optional):** The availability of the device associated with this capability.
  - **preferredEndpointId (optional):** Ignored.

2.2.2.7.4 userProperties

The server uses the **userProperties** category instance to store user information. The source of this information can be a corporate directory. The information is intended for use by the publisher.

The following example illustrates the format of a **userProperties** category instance.

```
<category name="userProperties" instance="0"
          publishTime="2008-01-23T07:50:12.873"
```

```
        container="1" version="7" expireType="static">

    <userProperties>
      <lines>
        <line lineType="Uc">
          tel:+11234567890;ext=67890
        </line>
      </lines>
      <telephonyMode>Uc</telephonyMode>
      <facsimileTelephoneNumber>+11234567891</facsimileTelephoneNumber>
      <streetAddress>1 Anywhere Street</streetAddress>
      <l>Somewhere</l>
      <st>SomeState</st>
      <countryCode>US</countryCode>
      <postalCode>12345</postalCode>
      <wWWHomePage>http://contoso.com</wWWHomePage>
      <exumEnabled>1</exumEnabled>
      <exumURL>
        EUM:bob@contoso.com;phone-context=EX-OCS-SIPSec.exchange.contoso.com
      </exumURL>
    </userProperties>
  </category>
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a **userProperties** document MUST conform to the schema defined in section 7.6.

- **userProperties:** Root element of the **userProperties** category. It has the following attributes:
  - **majorVersion (optional):** The major version of schema format.
  - **minorVersion (optional):** The minor version of schema format.
- **lines:** This element contains **line** subelements. This is a child element of the **userProperties** element.
- **line:** The value of this element MUST be a valid **tel** URI of the phone line associated with the user. The **line** element has the following attributes.
  - **lineType (required): lineType** MUST have one of the following tokens: "Uc", "UcPrivate", "Rcc", or "Dual".
- **lineServer (optional):** The URI of the line server that controls an RCC phone, for example, "sip:Server1@pbxgateway.contoso.com". This MUST be a valid SIP URI. This attribute MUST be set if **lineType** is "Rcc". This attribute is optional if **lineType** is "Dual". This attribute SHOULD NOT be set if the **lineType** is "Uc". This attribute MUST be set if **lineType** is "UcPrivate".
- **telephonyMode:** The value of this element controls the audio/video features on the SIP protocol client. It SHOULD<16> have one of the following tokens: "None", "Uc", "Rcc", "Dual" or "NoAudioVideo".
- **facsimileTelephoneNumber:** The user's fax number. This information SHOULD be retrieved from the DS.
- **streetAddress:** The user's street address. This information SHOULD be retrieved from the DS.
- **l:** The name of the user's city. This information SHOULD be retrieved from the DS.

- **st:** The name of the user's state. This information SHOULD be retrieved from the DS.
- **countryCode:** The name of the user's country code. Its value MUST be a two-character country code based on [ISO-3166]. This information SHOULD be retrieved from the DS.
- **postalCode:** The name of the user's postal code. This information SHOULD be retrieved from the DS.
- **wWWHomePage:** The URL of the user. This information SHOULD be retrieved from the DS.
- **exumEnabled:** The value of the first bit indicates whether the voice mail for a user is configured, 1 or 0. The SIP protocol client enables or disables voice mail features based on the value of the first bit.
- **exumURL:** The URL for Unified Messaging (UM). UM is the server role that enables an email server to serve as the voice mail system for a SIP server that implements [MS-SIPREGE].
- **forwardingUrls:** This element MUST be ignored by the SIP protocol client.
- **acpInformation:** This element contains information about **PSTN** bridging provided by third-party providers. This element has an attribute **default** of type Boolean which, if set to **true**, indicates that this is the default audio conferencing provider (ACP) information. Note that there can be multiple **acpInformation** elements, but only one of them can have **default** as **true**. The subelements are as follows:

- **tollNumber (string):** The toll number.
- **tollFreeNumber (string):** The toll-free number, if any.
- **participantPassCode (string):** Numbers that are the participant pass code used by **participants** to join the meeting.
- **domain (string):** The name of ACP provider's domain.
- **name:** Name of the ACP provider.
- **url:** Link to the **Web site** of the ACP provider.

Schema definition of this element is shown in the following example. The tollNumber and participantPassCode combination MUST be unique if multiple **acpInformation** elements exist.

The following example is XML for **acpInformation**:

```
<acpInformation default="true">
 <tollNumber>+14255550100</tollNumber>
 <tollFreeNumber>+1800xxxxxxx</tollFreeNumber>
 <tollFreeNumber>+1800xxxxxxx</tollFreeNumber>
 <participantPassCode>123xx</participantPassCode>
 <domain>acpbridge.contoso.com:4443</domain>
 <name>Contoso audio conferencing provider</name>
 <url>https://help.contoso.com</url>
</acpInformation>
<acpInformation default="false">
 <tollNumber>+14255550102</tollNumber>
 <tollFreeNumber>+1800xxxxxxx</tollFreeNumber>
 <tollFreeNumber>+1800xxxxxxxx</tollFreeNumber>
 <participantPassCode>1xx2345</participantPassCode>
 <domain>acpbxxridge.contoso.com</domain>
 <name>xxx provider</name>
 <url>https://xxxhelp.contoso.com</url>
</acpInformation>
<acpInformation default="false">"
```

```
 <tollNumber>+14255550103</tollNumber>
 <participantPassCode>12xxxx</participantPassCode>
 <domain>acpbridge2.contoso.com</domain>
 <name>provider3</name>
</acpInformation>
```

2.2.2.7.5 contactCard

A **contactCard category** instance contains user information such as a job title, phone number, and address that is shared with subscribers.

The following examples illustrate the format of a **contactCard** category instance:

```
<contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
  <identity>
    <name>
      <displayName>Bob</displayName>
    </name>
  </identity>
</contactCard>

<contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
  <phone type="work">
    <uri>tel:1234567890;phone-context=dialstring</uri>
  </phone>
  <phone type="mobile">
    <uri>tel:1234567890;phone-context=dialstring</uri>
  </phone>
</contactCard>

<contactCard
xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard"isUCEnabled="true">
</contactCard>

<contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
    <identity>
        <name>
            <displayName>HumanResource</displayName>
        </name>
    </identity>
    <automaton>true</automaton>
    <type>huntgroup</type>
    <description>Human Resources Group</description>
</contactCard>
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a contactCard document MUST conform to the schema defined in section 7.3

- **contactCard:** This is a root element of the **contactCard** category instance. This element has the following attributes.
  - **majorVersion (optional):** Major version of contact card format.
  - **minorVerson (optional):** Minor version of contact card format.
  - **isUCEnabled(optional):** A Boolean value that indicates whether the user is enabled for**Unified Communications**.<17>
- **identity:** The identity information of a contact card. This is a subelement of the **contactCard** element.

- **name:** The name of a user. This is a subelement of the **identity** element.
- **displayName:** A user's display friendly name. This is the subelement of the **name** element. It MAY have the following attributes:

- **LCID (optional):** Locale identifier of the **displayName**.
- **updated (optional):** The last time that the **displayName** element was updated, expressed in Coordinated Universal Time (UTC), as specified by [ISO-8601].

- **email:** The user's e-mail address. This is the subelement of the **identity** element.
- **address:** The user's postal address. This is the subelement of the **contactCard** element. It MAY have the following attribute:
- **type (optional):** Type of the address, which MUST be "work", "home", or "other".
- **street:** The user's street address. This is the subelement of the **address** element. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier of the street name.
  - **updated (optional):** The last time that the **street** element was updated, expressed in UTC, as specified by [ISO-8601].
- **city:** The user's city name. This is the subelement of the **address** element. It MAY have the following attributes.
  - **LCID (optional):** Locale identifier of the city name.
  - **updated (optional):** The last time that the **city** element was updated, expressed in UTC, as specified by [ISO-8601].
- **state:** The user's state name. This is the subelement of the **address** element. It MAY have the following attributes.
  - **LCID (optional):** Locale identifier of the state name.
  - **updated (optional):** The last time that the **state** element was updated, expressed in UTC, as specified by [ISO-8601].
- **zipcode:** The user's postal code. This is the subelement of the **address** element. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier of the postal code.
  - **updated (optional):** The last time that the **zipcode** element was updated, expressed in UTC as specified by [ISO-8601].
- **countryCode:** The user's country code. This is the subelement of the **address** element. Its value MUST be a two-character country code based on [ISO-3166].
- **company**: The name of the user's company. This is the subelement the of **contactCard** element. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier of the company name.
  - **updated (optional):** The last time that the **company** element was updated, expressed in UTC, as specified by [ISO-8601].
- **department:** The name of the user's department. This is the subelement of the **contactCard** element. It MAY have the following attributes:

- **LCID (optional):** Locale identifier of the department name.
- **updated (optional):** The last time that the **department** element was updated, expressed in UTC, as specified by [ISO-8601].

- **title:** The user's title. This is the subelement of the **contactCard** element. It MAY have the following attributes.
  - **LCID (optional):** Locale identifier of the title name.
  - **updated (optional):** This contains the last update time, in UTC, of the **title** element.
- **office:** A description of the user's office. This is the subelement of the **contactCard** element. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier of the office description.
  - **updated (optional):** The last time that the **office** element was updated, expressed in UTC.
- **url:** The URL of the user. This is the subelement of the **contactCard** element. This element MAY have the following attributes:
  - **type (optional):** Type of the URL. The value MUST be "sharepoint", "voicemail", or "other".
  - **updated (optional):** The last time that the **url** element was updated, expressed in UTC.
- **phone:** The user's phone number. This is the subelement of the **contactCard** element. This element MAY have the following attributes:
  - **type (optional):** Type of the phone, which SHOULD be "work", "home", "mobile", "other", or "custom1".
  - **updated (optional):** The last time that the **phone** element was updated, expressed in UTC.
- **uri:** The normalized **tel** URI format of a phone number. This is the subelement of the **phone** element. It MAY have the following attribute:
  - **updated (optional):** The last time that the **uri** element was updated, expressed in UTC.
- **displayString:** The display name of a phone number. This is the subelement of the **phone** element. It MAY have the following attributes:
  - **LCID (optional):** Locale identifier of the **displayString**.
  - **updated (optional):** The last time that the **displayString** element was updated, expressed in UTC.

- **automaton:** A Boolean value that indicates whether the **presentity** is an automated agent. If the **presentity** is an automated agent, the value SHOULD be set to **true**. If the **presentity** is not an automated agent, either the value SHOULD be set to **false** or the **automaton** element SHOULD be omitted. An automated agent is also known as a **bot**. An automated agent is a self-operating **presentity** that provides automatic responses to requests through instant messages, voice and other modes. This is the subelement of the **contactCard** element.<18>
  - **type:** The type of the **presentity**. This is the subelement of the **contactCard** element. It MUST have one of the following values:

- **person:** The **presentity** is an end user.
- **huntgroup:** The **presentity** distributes phone calls from a caller to a group of agents without interacting with the caller.
- **autoattendant:** The **presentity** distributes phone calls from a caller to a group of agents after interacting with the caller first.
- **automaton:** The **presentity** is a self-operating **presentity** that provides automatic responses to requests through instant messages.

- **description:** The description of **presentity**. This is the subelement of the **contactCard** element.<19>

displayADPhoto:<20> A Boolean value that controls whether a user picture is displayed. If the value is **true**, then the source of the displayed picture is the value of the **photo** subelement or the picture (if one exists) from the **Active Directory**. If the value is **false**, no picture is displayed. This is the optional subelement of the **contactCard** element.

- **photo:**<21> The information about the photo of the **presentity**. There can be zero or more **photo** subelements in a **contactCard** element. This element SHOULD have the following attribute:

- **type:** Type of the photo. It SHOULD have one of the tokens in the following table.

Values	Meaning
default	Default photo.
enterprise	Workplace photo.
exchange	E-mail server photo.
other	This is some other photo.

- **uri:**<22> The photo URI of the **presentity**. This is the optional subelement of the **photo** element. This element SHOULD have the following attribute:
  - **updated (optional):** The last time that the **uri** element was updated, expressed in UTC.
- **hash:**<23> A **cyclic redundancy check (CRC) hash** of **photo** if the **type** of the **photo** is "default" or "enterprise"; an email server ChangeKey if the **type** of the **photo** is "exchange". This is the mandatory subelement of the **photo** element.

2.2.2.7.6 legacyInterop

The server uses this category to generate the NOTIFY for PIDF and MSRTC subscriptions. The server aggregates the instances of the **state** category to compute a **legacyInterop** category instance as described in section 3.8.5. This **category** MUST be registered as a private category.

The following example illustrates the format of a **legacyInterop category** instance.

```
<legacyInterop xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
               availability="8400" token="urgent-interruptions-only" />
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a legacyInterop document MUST conform to the schema defined in section 7.7.

legacyInterop: This is the root element of the **legacyInterop** category instance. It has the following attributes:

- **availability (required):** This attribute contains a number that represents the availability of a presence state. For details about the **availability** element, see section 2.2.2.7.1.
- **token (optional):** Localizable token that represents the activity string. For details about the **token** attribute, see section 2.2.2.7.1.

- **dndState (optional):** If true, this attribute indicates that the **dndState category** instance contains the "Do Not Disturb" state of the **self-user**. For details about the **dndState category**, see section 2.2.2.7.10.

Even though this **category** is private, notifications are generated when the category is subscribed to through self SUBSCRIBE.

2.2.2.7.7 routing

The **routing** category MUST be registered as a private category. For details about the format of this category, see [MS-SIPRE] section 2.2.9.

2.2.2.7.8 calendarData

A **calendarData** category instance contains calendar information for a user, such as a free busy time slots, and working hours that are used for call routing and shared with subscribers.

The following examples illustrate the format of a **calendarData** category instance:

```
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
      <WorkingHours
          xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
        <TimeZone>
          <Bias>480</Bias>
          <StandardTime>
            <Bias>0</Bias>
            <Time>02:00:00</Time>
            <DayOrder>1</DayOrder>
            <Month>11</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </StandardTime>
          <DaylightTime>
            <Bias>-60</Bias>
            <Time>02:00:00</Time>
            <DayOrder>2</DayOrder>
            <Month>3</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </DaylightTime>
        </TimeZone>
        <WorkingPeriodArray>
          <WorkingPeriod>
            <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
            <StartTimeInMinutes>480</StartTimeInMinutes>
            <EndTimeInMinutes>1020</EndTimeInMinutes>
          </WorkingPeriod>
        </WorkingPeriodArray>
      </WorkingHours>
    </calendarData>

    <calendarData
          xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
      <freeBusy
          startTime="2009-10-22T07:00:00Z"
```

```
            granularity="PT15M" encodingVersion="1">
        </freeBusy>
    </calendarData>
```

Elements and attributes are defined in this section. For the full XML schema, see section 7.

The XML elements of a **calendarData** document MUST conform to the schema defined in section 7.8.

- **calendarData:** This is a root element of the **calendarData** category instance. A **calendarData** element consists of a sequence of zero or one **WorkingHours** or **freeBusy** element. The **calendarData** element has the following attributes:
  - **majorVersion (optional):** Major version of calendar data format.
  - **minorVersion (optional):** Minor version of the calendar data format.
  - **mailboxID (required):** **SMTP** address of the user's mailbox.
- **WorkingHours:** Contains an array of working hours in a given time zone. This is the subelement of the **calendarData** element.
- **TimeZone:** This element specifies a time-zone time. This is a subelement of the **WorkingHours** element.
- **Bias:** This element specifies time zone offset, representing the difference in minutes between the local time in this time zone and the UTC. This is the subelement of the **TimeZone** element.
- **StandardTime:** This element specifies a time-zoned time, representing the date and time in the contained time zone when time changes over to standard time. This is the subelement of the **TimeZone** element.
- **DaylightTime:** This element specifies time-zone time, representing the date and time in this time zone when time changes over to daylight time in the current year for Daylight Saving Time. This is a subelement of the **TimeZone** element.
- **Bias:** The time offset in minutes from the **Bias** value set on the parent **TimeZone** element to account for standard time or day light time in this time zone. This is a subelement of **DaylightTime** and **StandardTime** elements.
- **Time:** Specifies the time in the given time zone. This is a subelement of **DaylightTime** and **StandardTime** elements.
- **DayOrder:** This element specifies the day number of a month. This is a subelement of **DaylightTime** and **StandardTime** elements.
- **Month:** This element specifies the month number of a time. This is a subelement of **DaylightTime** and **StandardTime** elements.
- **DayOfWeek**: This element specifies the day of week. This is a subelement of **DaylightTime** and **StandardTime** elements.
- **WorkingPeriodArray:** This element contains an array of **WorkingPeriod** elements. This is a subelement of **WorkingHours** element.
- **WorkingPeriod:** This element specifies the working period details. This is a subelement of **WorkingPeriodArray** element.
- **DayOfWeek:** This element specifies the day of a week. This is a subelement of the **WorkingPeriod** element. The value of this element is a string composed of one or more of the following words:

- Sunday
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday
- Day
- Weekday
- WeekendDay

- **StartTimeInMinutes:** This element contains the start time of each workday after midnight in minutes. This is the subelement of the **WorkingPeriod** element.
- **EndTimeInMinutes:** This element contains the end time of each workday after midnight in minutes. This is the subelement of **WorkingPeriod** element.
- **freeBusy:** This element contains consecutive calendar blocks, showing free-busy intervals of a specified duration from a specified starting time. This is a subelement of **calendarData** element. The value of this element, is a **base64** encoded string on the stream of binary bits. The interpretation of the binary bits differs for different **calendarData** category instance:

If the two most significant bits of the instance **ID** value are "00":

Every 2 bits represent a calendar block of the duration defined by the granularity value in minutes. Data starts from the given **startTime** in UTC. Calendar block types are encoded as follows:

- 00, Free (Fr)
- 01, Tentative (Te)
- 10, Busy (Bu)
- 11, Out of facility (Oo)

For example, the **startTime** attribute value can be "2007-09-01T00:00:00Z". A granularity value of "PT15M" shows calendar blocks as fifteen-minute intervals. If the calendar blocks in are in the following order:

```
FrFrFrFrFrBuBuBuBuTeTeTeTeOoOoOoOoFrFr
```

This corresponds to the following bit stream:

000000000010101010010101011111111110000

If the two most significant bits of the instance ID value are 01:<24>

The first byte contains the maximum number of free/busy states. States are encoded as follows:

- 0, Free
- 1, Tentative

- 2, Busy
- 3, Out of facility
- 4, Working elsewhere
- 5-255, reserved for future use.

  Next, it's a bytes array and the size of the array equals to the maximum number of free/busy states minus five. This array is used as a fallback map: if a newer free/busy state cannot be interpreted, it is interpreted as the value in the corresponding entry in the fallback map; the index to be used is the number of the state minus five. The value in any position will be less than the index plus five to ensure the fallback will only happen from a newer state to an older state.

  Next, it's the bit stream containing the actual free/busy data. It SHOULD contain 384 entries and each entry represents a calendar block of the duration defined by the granularity value in minutes. The width of each entry MUST be the smallest width possible to hold the maximum number of the free/busy states. Data starts from the given **startTime** in UTC.

  For example, the **startTime** attribute value can be "2007-09-01T00:00:00Z". A granularity value of "PT15M" shows calendar blocks as fifteen-minute intervals. If the calendar blocks in are in the following order:

  ```
  Free Free Busy Busy Tentative Tentative Oof Oof WorkingElsewhere Free
  ```

  This corresponds to the following bit stream:

  ```
  000000010010001001011011100000
  ```

The free-busy data shown by the value of a **freeBusy** element is the base64 encoded value of the preceding binary bits.

This element has the following attributes:

- **startTime (required):** Starting time expressed in UTC.
- **granularity (required):** Duration of time slots in minutes showing free or busy on calendar.
- **encodingVersion (required):** Encoding version number of free busy data.

2.2.2.7.9 workingHours

This section describes behaviors supported in versions described by endnote.<25>

The **workingHours** category instance SHOULD contain working hours of the self-user to support call routing based on working hours, if enabled in the **routing** category. The server SHOULD generate a **workingHours** category instance from the **WorkingHours** element of the **calendarData** category instance, as specified in section 3.8.5.3. This category MUST be registered as a private category on the server.

The following examples illustrate the format of a **workingHours** category instance:

```
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
      <WorkingHours
         xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
        <TimeZone>
          <Bias>480</Bias>
```

```
        <StandardTime>
          <Bias>0</Bias>
          <Time>02:00:00</Time>
          <DayOrder>1</DayOrder>
          <Month>11</Month>
          <DayOfWeek>Sunday</DayOfWeek>
        </StandardTime>
        <DaylightTime>
          <Bias>-60</Bias>
          <Time>02:00:00</Time>
          <DayOrder>2</DayOrder>
          <Month>3</Month>
          <DayOfWeek>Sunday</DayOfWeek>
        </DaylightTime>
      </TimeZone>
      <WorkingPeriodArray>
        <WorkingPeriod>
          <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
          <StartTimeInMinutes>480</StartTimeInMinutes>
          <EndTimeInMinutes>1020</EndTimeInMinutes>
        </WorkingPeriod>
      </WorkingPeriodArray>
    </WorkingHours>
  </calendarData>
```

The format and schema of the **workingHours** category MUST be the same as the **calendarData** category instance, as defined in section 2.2.2.7.8. For the full XML schema, see section 7.

2.2.2.7.10 dndState

This section describes behaviors supported in versions described by endnote.<26>

The server MUST look at the **userState** instance and **presentingState** instance of the **state** category to generate the **dndState category** instance that contains the "Do Not Disturb" state of the **self-user**, as defined in section 3.8.5.4. This category MUST be registered on the server as a private category. The server SHOULD do call routing based on the **dndState** category instance if the **routing** category instance has enabled call routing.

The following examples illustrate the format of a **dndState** category instance.

The **dndState** instance data with "Do not disturb".

```
<state xsi:type="userState" manual="true"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://schemas.microsoft.com/2006/09/sip/state">
  <availability>9500</availability>
</state>
```

The **dndState** instance data without "Do not disturb".

```
<state xsi:type="userState" manual="true"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://schemas.microsoft.com/2006/09/sip/state">


   </state>
```

The format and schema of the **dndState** category SHOULD be the same as the **state** category, as defined in section 2.2.2.7.1. For the full XML schema, see section 7.

2.2.2.7.11 mwi

This section describes behaviors supported in versions described by endnote <27>.

The **mwi** category instance MAY contain voice mail information of the self-user. This category instance SHOULD contain unread voice mail count, read voice mail count and message waiting indication.

The following example illustrates the format of a **mwi** category instance.

```
<mwi
  xmlns="http://schemas.microsoft.com/2006/09/sip/mwi"
  messageWaiting="true"
  unreadVoiceMailCount="30"
  readVoiceMailCount="13" />
```

Elements and attributes are defined as follows. For the full XML schema, see section 7.

The XML elements of a **mwi** document MUST conform to the schema defined in section 7.9.

- **mwi:** A root element of the **mwi** category instance. This element has the following attributes.
  - **majorVersion (optional):** Major version of mwi data format.
  - **minorVersion (optional):** Minor version of the mwi data format.
  - **messageWaiting (required):** Boolean value that indicates whether there is a new voice mail for the self-user (**true**) or (**false**).
  - **unreadVoiceMailCount (optional):** Unread voice mail count for the self-user.
  - **readVoiceMailCount (optional):** Read voice mail count for the self-user.

2.2.2.7.12 linkedPICContacts

This section describes behaviors supported in versions described by endnote<28>.

The linkedPICContacts category instance contains the linked identities of the PIC contacts that represent the same presentity. The individual identities are received from the SIP Server through a in a NOTIFY response. The SIP protocol client replicates the data from the NOTIFY response into this category so that it can keep track of all the secondary identities for a given presentity. This category MUST be registered on the server as a private category.

The following example illustrates the format of a linkedPICContacts category instance.

```
<category name="linkedPICContacts" instance="0" publishTime="2015-01-27T21:39:25.927"
container="1" version="57" expireType="static">
<LinkedPicContactList>
        <LinkedPicContactEntry primary-Identity="bob@hotmail.com">
          <secondary-identities>
            <secondary-identity>bob@skypeids.net</secondary-identity>
            <secondary-identity>8d9a9d943984476c@skypecid.net</secondary-identity>
          </secondary-identities>
        </LinkedPicContactEntry>
        <LinkedPicContactEntry primary-Identity="alice@hotmail.com">
          <secondary-identities>
            <secondary-identity>alice@skypeids.net</secondary-identity>
            <secondary-identity>445abd8439841238@skypecid.net</secondary-identity>
          </secondary-identities>
        </LinkedPicContactEntry>
```

```
  </LinkedPicContactList>
</category>
```

Elements and attributes are defined as follows. For the full XML schema, see section 7.

The XML elements of a linkedPICContacts document MUST conform to the schema defined in section 7.10.

- **linkedPicContactList**: A root element of the linkedPICContacts category instance. This element will contain a list of LinkedPicContactEntry elements.
- **LinkedPicContactEntry**: A Root element of an identity entry in the category.
  - This element has the following attributes.
    - **primary-Identity**: URL representing primary identity on which this presentity should be contacted for all outgoing SIP Invites.
  - The **LinkedPicContactEntry** contains the following child elements.
- **secondary-identities**: A Root element of the secondary identities that belong to the same presentity . This element will contain a list of secondary-identity elements as follows.
  - **secondary-identity**: URL that belongs to the same presentity represented in the primary-Identity attribute. This URL should receive the same permissions as the primary-Identity. Incoming SIP Invites may be received from these URL by the presentity represented in the primary-Identity attribute. This Incoming SIP Invite should be given the same permission level as that represented in the primary-Identity attribute.

2.2.2.7.13 Persistent chat categories

If a SIP protocol client connects to a persistent chat server using [MS-XCCOSIP] protocol<29> this client MAY publish user settings and preferences that will be saved between user logon sessions and shared between multiple clients signed in by the same user. Persistent chat categories employ a **propertyList** XML type defined in section 7.1 which contains a collection of properties.

Elements and attributes are defined as follows.

propertyList: the root element of any persistent chat category that MUST contain one or more child **property** elements.

property: this element describes an individual property and has two attributes:

name: property name
type: property type (can be a string, numeric, Boolean or object)

and an element

value: property value.

2.2.2.7.13.1 roomSetting

A SIP protocol client MAY publish a list of chat channels (rooms) that the user is observing using a **roomSetting** category instance to enumerate properties of each room. The **roomSetting** category is illustrated by the following example:

```
<category name="roomSetting" instance="1131464332"
      publishTime="2012-06-13T22:19:51.470" container="1"
```

```
        version="1" expireType="static">
  <propertyList xmlns="http://schemas.microsoft.com/2010/12/sip/propertyList">
    <property name="ver" type="string"><value>0.3</value></property>
    <property name="url" type="string">
      <value>ma-chan://contoso.com/89353b48-48f0-44a5-8df8-ee372e4ed467</value>
    </property>
    <property name="name" type="string"><value>Test Room</value></property>
    <property name="setting" type="numeric"><value>33</value></property>
    <property name="poolUri" type="string">
      <value>sip:PersistentChatService@contoso.com</value>
    </property>
    <property name="poolName" type="string"><value>Europe</value></property>
  </propertyList>
  </category>
```

Elements and attributes are defined as follows.

The XML elements of a **roomSetting** document MUST conform to the schema defined in section 7.1

propertyList: the root element of the **roomSetting** category that MUST contain following child **property** elements:

Property Name	Property Type	Description
Ver	string	Format version
url	string	Room URL
name	string	Room name
setting	numeric	A logically ORed set of notification flags; see section 2.2.2.7.13.4 for flags' values
poolUri	string	URI of a pool of persistent chat servers
poolName	string	A friendly name of a pool of persistent chat servers

2.2.2.7.13.2 roomUpdate

Each message in a persistent chat system based on [MS-XCCOSIP] protocol has an integer id that uniquely identifies this message in a room it was posted to. This id is monotonically increasing, so remembering the id of the last read message allows a SIP protocol client to implement UI to visually distinguish unread messages. A SIP protocol client MAY publish such a last read message id for a room using **roomUpdate** category instance. The **roomUpdate** category is illustrated by the following example:

```
  <category name="roomUpdate" instance="1131464332"
        publishTime="2012-06-13T22:19:51.330" container="1"
        version="1" expireType="static">
  <propertyList xmlns="http://schemas.microsoft.com/2010/12/sip/propertyList">
    <property name="ver" type="string"><value>0.3</value></property>
    <property name="url" type="string">
      <value>ma-chan://contoso.com/89353b48-48f0-44a5-8df8-ee372e4ed467</value>
    </property>
    <property name="msgid" type="numeric"><value>12</value></property>
  </propertyList>
  </category>
```

Elements and attributes are defined as follows.

The XML elements of a **roomUpdate** document MUST conform to the schema defined in section 7.1

propertyList: the root element of the **roomUpdate** category that MUST contain following child **property** elements:

Property Name	Property Type	Description
Ver	string	Format version
url	string	Room URL
Msgid	numeric	Message id of the last read message in this room

2.2.2.7.13.3 roomInvitation

A user of a persistent chat system based on [MS-XCCOSIP] protocol can be invited to join certain rooms in the system. Each invitation is a protocol object and has an integer id that uniquely identifies it for the user. This id is monotonically increasing, so remembering the id of the last seen invitation allows a SIP protocol client to implement UI to visually distinguish new invitations. A SIP protocol client MAY publish such a last seen invitation id using **roomInvitation** category instance. The **roomInvitation** category is illustrated by the following example:

```
<category name="roomInvitation" instance="4126539244"
        publishTime="2012-06-01T22:18:07.973" container="1"
        version="1" expireType="static">
<propertyList xmlns="http://schemas.microsoft.com/2010/12/sip/propertyList">
  <property name="Version" type="string"><value>0.4</value></property>
  <property name="Pool" type="string">
    <value>sip:PersistentChatService@contoso.com</value>
  </property>
  <property name="LastId" type="numeric"><value>69</value></property>
</propertyList>
```

Elements and attributes are defined as follows.

The XML elements of a **roomInvitation** document MUST conform to the schema defined in section 7.1

propertyList: the root element of the **roomInvitation** category that MUST contain following child **property** elements:

Property Name	Property Type	Description
Version	string	Format version
Pool	string	URI of a pool of persistent chat servers
LastId	numeric	Id of the last seen invitation

2.2.2.7.13.4 gcFilterSetting

A SIP-protocol client in a persistent chat system based on [MS-XCCOSIP] protocol CAN create filters to monitor room conversations and notify the user when a message with certain keywords is posted to any of the observed rooms. A SIP protocol client MAY publish such filter properties using a **gcFilterSetting** category instance. The **gcFilterSetting** category is illustrated by the following example:

```
<category name="gcFilterSetting" instance="1131464364"
        publishTime="2012-06-13T22:21:14.153" container="1"
        version="1" expireType="static">
```

```
<propertyList xmlns="http://schemas.microsoft.com/2010/12/sip/propertyList">
  <property name="ver" type="string"><value>0.4</value></property>
  <property name="notificationSettings" type="numeric">
    <value>33</value>
  </property>
  <property name="type" type="numeric"><value>0</value></property>
  <property name="name" type="string"><value>TestFilter</value></property>
  <property name="senderscope" type="numeric"><value>0</value></property>
  <property name="specificsenders" type="string"><value></value></property>
  <property name="Keywords" type="string"><value>Test Run</value></property>
  <property name="MatchWholeWords" type="boolean"><value>true</value></property>
  <property name="MatchAllWords" type="boolean"><value>false</value></property>
  <property name="filterId" type="string">
    <value>{FCDB5A83-9E50-4EBC-BDAB-4468DF017776}</value>
  </property>
</propertyList>
</category>
```

Elements and attributes are defined as follows.

The XML elements of a **roomSetting** document MUST conform to the schema defined in section 7.1

propertyList: the root element of the **roomSetting** category that MUST contain following child **property** elements:

Property Name	Property Type	Description
ver	string	Format version
notificationSettings	numeric	A logically ORed set of notification flags
type	numeric	Filter type, MUST be set to 0
name	string	Filter name
senderscope	numeric	Instructs the filter to monitor messages from certain senders: 0 - anyone except me; 1 - anyone; 2 - specific senders.
specificsenders	string	A semicolon-separated list of SIP URIs of specific senders. SHOULD be non-empty if *senderscope* is set to 2.
Keywords	string	Filter keywords, separated by whitespace
MatchWholeWords	boolean	Match entire words
MatchAllWords	boolean	Match all keywords. For example, if Keywords is "Test Run" and MatchAllWords is set, the filter will be looking for both "Test" and "Run" to be present in the message. If this property is not set the filter will look for either "Test" or "Run".
filterId	string	A unique filter identifier, GUID.

The following notification flags for filters and rooms are defined:

0x1	Use global rather than per room/filter settings
0x2	Open a separate room window on a new message arrival
0x4	Display a toast on a new message arrival
0x8	Play sound on a new message arrival
0x10	Open a separate room window on a new high-importance message arrival
0x20	Display a toast on a new high-importance message arrival

0x1	Use global rather than per room/filter settings
0x40	Play sound on a new high-importance message arrival

2.2.2.8 PIDF/RPID Subscription Related Messages

The server MUST support receiving a SUBSCRIBE request for presence in PIDF format with RPID extensions, as defined in [RFC3863] section 4.1 and [RFC4480] section 5. Support for this capability on the server is required primarily for interoperability with federated clouds.

The server MUST NOT support publication in PIDF format. If a publication in PIDF format is received, the server will not return any response. When presence information is published in enhanced presence, the server's conversion logic MUST map the presence published in enhanced presence to PIDF format.

2.2.2.8.1 PIDF/RPID Subscription Request Message

PIDF/RPID subscription uses SUBSCRIBE request and response messages, as defined in [RFC3265] section 3.1 and section 3.2, with the presence event package. The **Event** header MUST be set to "presence".

The **Accept**, **Supported, Require**, and **Proxy-Require** header fields indicate support for a variety of presence extensions that are described in [MS-SIP] section 3.2.

The **Accept** header field indicates that the SIP protocol client is capable of receiving, in response to SUBSCRIBE, a presence document ("application/xpidf+xml , text/xml+msrtc.pidf, application/pidf+xml"). The **Accept** header field MUST include "application/pidf+xml".

The **From-URI** MUST be the SIP URI of the subscriber. The **To-URI** MUST be the SIP URI of the presence publisher.

Require headers MUST NOT be present in the request.

The SIP protocol client MUST add a valid **Contact** header that is used as a remote target URI of the SIP dialog route set as specified in [RFC3261] section 12.

In the following sample, a federated user, alice@fabrikam.com, subscribes to the presence information from bob@contoso.com according to the semantics of [RFC3265] section 3.1, with an **Event** header field value of "presence" and an **Accept** header field value of "application/pidf+xml".

```
SUBSCRIBE sip:bob@contoso.com;maddr=server.contoso.com SIP/2.0
From: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
To: <sip:bob@contoso.com>
CSeq: 1 SUBSCRIBE
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Max-Forwards: 67
Contact: <sip:server.fabrikam.com:5061;maddr=server.fabrikam.com;transport=TLS>
Expires: 16406
Accept: application/pidf+xml
Event: presence
Content-Length: 0
```

The server sends a NOTIFY request with the presence of the user, bob@contoso.com. The server MUST set the **Content-Type** header field to "application/pidf+xml". The server does not support PIDF/RPIF data formats, defined in [RFC3863] section 4.1 and [RFC4480] section 5, in their entirety, but only the subset required for interoperability with federated presence clouds.

The PIDF document contains a single PIDF tuple that contains attributes from basic PIDF, as well as the following attributes from Rich Presence Extensions to PIDF (RPID) and Contact Information in PIDF (CIPID) extensions:

- **Available/activity**: The activity is an RPID extension that allows SIP protocol clients that provide presence information, called presentities, to attach information about what the presentity is actually doing.
- **display-name**: A CIPID attribute that provides the user's friendly name.

The PIDF document MAY also contain the following elements that represent the multiple identities associated with the same person.

- **cid** – Represents the unique identifier for this user. This identifier will be used to by the client to fetch photos of the user by crafting a URL .
- **linked-identities**: Root element that contains the list of secondary URLs associated with the same person. These identities should be given the same permissions as the primary Identity.
  - This element contains the following attributes.
    - **primary-id**: This URL RL should be the preferred identity that is used for all outgoing SIP Invites.
  - The **linked-identities** contains the following child elements.
    - **secondary-identities**: A Root element of the secondary identities that belong to the same presentity. This element will contain a list of secondary-identity elements as follows.
- **secondary-identity**: URL that belongs to the same presentity represented in the primary-Identity attribute. This URL should receive the same permissions as the primary-Identity. Incoming SIP Invites may be received from these URL by the presentity represented in the primary-Identity attribute. This Incoming SIP Invite should be given the same permission level as that represented in the primary-Identity attribute.

The server passes these attributes in a single PIDF tuple, as shown in the following example:

```
NOTIFY sip:server.fabrikam.com:5061;maddr=server.fabrikam.com;transport=TLS SIP/2.0
From: <sip:bob@contoso.com>;tag=6667115B
To: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
CSeq: 1 NOTIFY
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Max-Forwards: 70
Content-Length: …
Content-Type: application/pidf+xml
Event: presence
subscription-state: active;expires=16406
…
<?xml version="1.0" encoding="utf-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
xmlns:ep="urn:ietf:params:xml:ns:pidf:status:rpid-status"
xmlns:et="urn:ietf:params:xml:ns:pidf:rpid-tuple"
xmlns:ci="urn:ietf:params:xml:ns:pidf:cipid" entity="sip:bob@contoso.com">
  <tuple id="0">
    <status>
      <basic>open</basic>
      <ep:activities>
        <ep:activity>away</ep:activity>
      </ep:activities>
    </status>
  </tuple>
  <linked-identities primary-id="bob@hotmail.com">
    <secondary-identities>
      <secondary-identity>bob@skypeids.net</secondary-identity>
```

```
      <secondary-identity>A1519ABC8FAC5FA71@skypecid.net</secondary-identity>
    </secondary-identities>
  </linked-identities>
  <cid>A1519ABC8FAC5FA71</cid>
  <ci:display-name>Bob at Contoso</ci:display-name>
</presence>
```

2.2.2.8.2 PIDF/RPID Subscription Response Message

If the operation is successful, the server MUST return a 200 OK response

The server sends a 200 OK acknowledging the receipt of the subscription from federated user, alice@fabrikam.com.

```
SIP/2.0 200 OK
From: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
To: <sip:bob@contoso.com>;tag=6667115B
CSeq: 1 SUBSCRIBE
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Expires: 16406
Content-Length: 0
```

The subscribing SIP protocol client acknowledges the receipt of the notification with a 200 OK, as shown in the following example:

```
SIP/2.0 200 OK
From: <sip:bob@contoso.com>;tag=6667115B
To: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
CSeq: 1 NOTIFY
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Content-Length: 0
```

2.2.2.9 MSRTC Subscription-Related Messages

The server MUST NOT support publication in MSRTC format. If a publication is received in MSRTC format, the server does not return a response. The server uses the conversion logic specified in section 3.7.5.5 to map the presence information published in enhanced presence to MSRTC format.

2.2.2.9.1 MSRTC subscription request

In the following example of an MSRTC subscription request, a federated user, alice@fabrikam.com, subscribes to presence information published by bob@contoso.com.

```
SUBSCRIBE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TLS 10.46.164.196:1620
Max-Forwards: 70
From: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
To: <sip:bob@contoso.com>
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 1 SUBSCRIBE
Contact: <sip:alice@fabrikam.com:1620;maddr=10.46.164.196;transport=tls>;proxy=replace
User-Agent: LCC/1.3
Event: presence
Accept: application/xpidf+xml, text/xml+msrtc.pidf, application/pidf+xml
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: ...
```

```
Content-Length: 0
```

In this example, an MSRTC subscription uses the SUBSCRIBE request and response as defined in [RFC3265] section 3.1 and section 3.2, with the presence event package. The **Event** header MUST be set to "presence".

The **Accept**, **Supported**, **Require**, and **Proxy-Require** header fields indicate support for a variety of presence extensions that are described in [MS-SIP] section 3.2.

The **Accept** header field indicates that the SIP protocol client is capable of receiving (in response to SUBSCRIBE) a presence document ("application/xpidf+xml , text/xml+msrtc.pidf, application/pidf+xml"). An **Accept** header field MUST include "text/xml+msrtc.pidf".

The From-URI MUST have the SIP URI of the subscriber. The To-URI MUST have the SIP URI of the presence publisher.

The SIP protocol client MUST add a valid **Contact** header field that can be used as a remote target URI of the SIP dialog route set as specified in [RFC3261] section 20.10.

2.2.2.9.2 MSRTC subscription response

The server sends a 200 OK acknowledging receipt of the subscription.

```
SIP/2.0 200 OK
Authentication-Info: ...
ms-edge-proxy-message-trust: ms-source-type=AutoFederation;ms-ep-fqdn=lcsproxy-
internal.fabrikam.com;ms-source-verified-user=verified;ms-source-network=federation
Contact: <sip:ocsserver.contoso.com:5061;transport=tls>
Content-Length: …
From: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
To: <sip:bob@contoso.com>;tag=B8FCF750
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 10.46.164.196:1620;ms-received-port=1620;ms-received-cid=1dcee00
Expires: 29087
Content-Type: text/xml+msrtc.pidf
Event: presence
subscription-state: active;expires=29087
ms-piggyback-cseq: 1
Supported: ms-piggyback-first-notify
Record-Route: ...

<presentity uri="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2002/09/sip/presence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <availability aggregate="300"/>
  <activity aggregate="400"/>
  <displayName displayName="Bob"/>
  <email email="bob@contoso.com"/>
  <phoneNumber number="+1 (123) 4567890 X67890"/>
  <aggregate>
    <states>
      <state avail="3500" xsi:type="userState">online</state>
    </states>
  </aggregate>
</presentity>
```

The server sends a NOTIFY request with the presence document when the presence of bob@contoso.com changes.

```
NOTIFY sip:10.46.164.196:1620;transport=tls;ms-received-cid=1DCEE00 SIP/2.0
Authentication-Info: Kerberos
rspauth="602306092A864886F71201020201011100FFFFFFFF01CD2AED53D3153FE91637B26C4C3F4E",
srand="03854A6C", snum="201", opaque="D22EB6E6", qop="auth",
targetname="sip/NALCSFE02.na.fabrikam.com", realm="SIP Communications Service"
Via: ...
To: <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
Content-Length: …
From: <sip:bob@contoso.com>;tag=B8FCF750
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 2 NOTIFY
Content-Type: text/xml+msrtc.pidf
Event: presence
subscription-state: active;expires=27503

<presentity uri="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2002/09/sip/presence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <availability aggregate="300"/>
  <activity aggregate="600"/>
  <displayName displayName="Bob"/>
  <email email="bob@contoso.com"/>
  <phoneNumber number="+1 (123) 4567890 X67890"/>
  <aggregate>
    <states>
      <state avail="6500" xsi:type="userState">busy</state>
    </states>
  </aggregate>
</presentity>
```

The server MUST set the **Content-Type** header field to "text/xml+msrtc.pidf".

The XML body that is generated MUST be a valid http://schemas.microsoft.com/2002/09/sip/presence document. For details about the format of this XML, see [MS-SIP] section 2.2.1. The http://schemas.microsoft.com/2002/09/sip/presence document is extended to include the **aggregate** element.

Elements and attributes of the **aggregate** element are defined in this section. For the full XML schema of the **aggregate** element, see section 9.

aggregate: Contains a **states** element. There are no specific attributes.

states: Contains **state** elements. There are no specific attributes.

state: This element represents a user's overall aggregated presence state. The value of this element contains a localizable token that represents the activity string of the state. For details, about the **token** attribute, see section 2.2.2.7.1. The **state** element has the following attributes:

avail (required): This attribute contains a number representing the availability of a presence state. For details, see the specification for the **availability** element in section 2.2.2.7.1.

xsi:type (required): The type of the **state** element. It MUST have the value **userState** to indicate that this state element represents a user's overall presence state.

The subscribing protocol client acknowledges receipt of the notification with a 200 OK.

```
SIP/2.0 200 OK
Via: ...
From: <sip:bob@contoso.com>;tag=B8FCF750
To: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 2 NOTIFY
User-Agent: LCC/1.3
```

```
Proxy-Authorization: ...
Content-Length: 0
```

2.2.2.10 Delegation-Related Messages

The following sections specify delegation-related messages for enhanced presence.

2.2.2.10.1 delegates Document

The following example illustrates the **delegates** document XML schema.

```
<delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="2">
    <delegate uri="john@contoso.com" publish="false" redelegate="false"/>
</delegates>
```

Elements and attributes of the delegates document are defined in this section. For the full XML schema of the delegates document, see section 6.5.

delegates: This element is a wrapper for a list of delegates. Multiple instances of the **delegate** element can exist in a single document. This element has the following attribute.

> **version (required):** This unsigned integer specifies the version number of the delegate list as maintained by the server for this **presentity**. The version number allows multiple endpoints of the same **presentity** to synchronize their change requests to the delegate list and allows them to decide when to accept a notification from the server.

delegate: This element contains information about one delegate. It has the following attributes.

uri (required): The SIP URI of the delegate.

publish (required): A Boolean attribute. The value of this attribute depends on the setup of the delegate relation, as described in 2.2.2.10.2. The value MUST be **true** if the **authorization** attribute of **publishOperation** element is set to **allow** in the **setDelegate** request and it MUST be **false** if authorization is set to **deny**.

redelegate (required): A Boolean attribute. The value of this attribute depends on the setup of the delegate relation, as described in 2.2.2.10.2. The value MUST be **true** if the **authorization** attribute of **redelegateOperation** element is set to **allow** in the **setDelegate** request and it MUST be **false** if authorization is set to **deny**.

2.2.2.10.2 setDelegate Request

A **presentity** uses the **setDelegate** request to set up the delegation relation between self and another user. The following example illustrates the format of such a message.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu="">
User-Agent: UCCP/3.0.6789.0 OC/3.0.6789.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-setDelegate +xml
Content-Length: …
```

```
<setDelegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegate-management"
version="1">
  <delegate uri="sip:john@contoso.com" action="add">
    <publishOperation authorization="deny"/>
    <redelegateOperation authorization="deny"/>
  </delegate>
</setDelegates>
```

The **Content-Type** header field MUST have a value of "application/msrtc-setDelegate+xml", which means that this SERVICE carries an XML body that contains a delegation request.

Elements and attributes of a **setDelegate** request are defined in this section. For the full XML schema for a **setDelegate** request, see section 6.14.

setDelegates: The root node for the request. It has the following attribute.

version (required): An unsigned integer that specifies the version number of the delegate list for the delegator.

delegate: Contains information regarding the delegate. It has the following attributes.

uri (required): The SIP URI of the delegate.

action (required): The action to perform. It can have one of the following values:

- add
- remove
- modify

It MUST be supported by the server following the description as follows:

- **add:** The specified delegate is a new delegate being added.
- **modify:** The specified delegate is a modification of an existing delegate.
- **remove:** The specified delegate is being removed from the list of delegates. The request MUST fail if there is no such existing delegate.

publishOperation (optional): This is an optional element. Specifies if the delegate can publish on behalf of the delegator.

authorization(required): It can have one of the following values:

- allow
- deny

redelegateOperation (optional): This is an optional element. Specifies if the delegate can redelegate on behalf of the delegator.

authorization (required): It can have one of the following values:

- allow
- deny

2.2.2.10.3 Response to a setDelegate Request

If the **setDelegate** request is successful, the server sends back a 200 OK response with a piggyback notification that contains the complete list of the delegates for the user. The following example illustrates the format of such a message:

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms
received-cid=B00
Content-Type: application/vnd-microsoft-roaming-self+xml

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
 xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
 xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
 xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
 xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="2">
    <delegate uri="john@contoso.com" publish="false" redelegate="false"/>
  </delegates>
</roamingData>
```

The **Content-Type** header field MUST have a value of "application/vnd-microsoft-roaming-self+xml", which means that this 200 OK carries an XML body that contains a roaming-self response that is fully specified in section 2.2.2.3.2.

Elements and attributes of a response to a **setDelegate** request are defined in section 2.2.2.10.1.

The XML elements of a **delegates** document MUST conform to the schema defined in section 6.5.

2.2.2.10.4 Version Conflict Response

If the **setDelegate** request fails because the version number specified on the delegates instance in the setDelegate request is not equal to the current version number at the server, the server sends back a 409 Conflict response. The body of this message contains information about the version number specified in the request and the current version number at the server. The following example illustrates the format of such a message.

```
SIP/2.0 409 Conflict
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>;tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>;tag=69DECB7060C5CD2546F1004F96625037
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms
received-cid=B00
ms-diagnostics: 2141;reason="Delegate version is out of date";source="..."
Content-Type: application/msrtc-fault+xml

<Fault>
  <Faultcode>Client.BadCall.WrongDelta</Faultcode>
  <details>
    <operation index="1" version="1" curVersion="2" >
    </operation>
  </details>
</Fault>
```

The ms-diagnostics header field is documented in [MS-OCER] section 2.2.1.1 and provides information that can be useful in further diagnostics. The **Content-Type** header field MUST have a value of "application/msrtc-fault+xml", which means that this 409 Conflict carries an XML body that contains an **msrtc-fault** response.

Elements and attributes of a 409 Conflict response are defined in this section.

Fault: Wraps the fault information.

Faultcode: The reason for the fault. A value of **Client.BadCall.WrongDelta** is used to indicate the version number in the publish request is not equal to the current version number at the server.

details: A wrapper for a list of faulting operations.

operation: Each operation element describes an operation that resulted in the fault. It has the following attributes:

> **index (required):** This unsigned integer identifies the **delegate** instance in the triggering **setDelegate** request.
>
> **version (required):** This unsigned integer returns the version number specified on the **delegates** instance in the **setDelegate** request.
>
> **curVersion (required):** This unsigned integer specifies the current version number of the **delegates** list instance maintained at the server. For a **setDelegate** request to be successful, it SHOULD be retried with this value.

3 Protocol Details

3.1 Directory Search Request Details

This section describes behaviors supported in versions described by endnote <30>.

3.1.1 Abstract Data Model

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

The server MUST maintain all of the state required to perform a **Lightweight Directory Access Protocol (LDAP)** lookup into the directory service (DS).

3.1.2 Timers

None.

3.1.3 Initialization

The server SHOULD initialize all connectivity logic to a DS and SHOULD initialize any XML engine that can be used for the parsing of the XML.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Processing a SERVICE Request

A user uses a SERVICE request to perform a DS user search. SERVICE SIP requests MUST have the following SIP fields set as specified:

- The **Content-Type** header field MUST be set to "application/SOAP+xml".
- The **From** header MUST be set to the URI of the user issuing the search request.
- The **To** header MUST be set to the SIP domain in which the search is being performed.
- A SIP request MUST NOT contain any **Require** headers.
- A SIP request MUST contain a body.

An example of a DS user search formatted as a SERVICE SIP request and response is found in section 4.1.

If the SERVICE request for the search is successful, the server sends back a 200 OK response with the results of the search. The Content-Type header of the XML body is "application/SOAP+xml", and the body itself is a SOAP XML response that adheres to the schema in section 8.

3.1.5.2 Performing the Search

The server uses an LDAP interface to perform a search against a DS.

The search is performed against a DS using the **Array** of search filters supplied in the SERVICE request to supply search criteria. The server takes all the filters specified and performs a logical AND search using their values.

The number of results that the server returns to the protocol client is limited by the lower of the **maxResults** value supplied by the protocol client and the maximum value set by a server administrator. If the number of results returned by the DS exceeds the number of results to be returned to the SIP protocol client, the **moreAvailable** attribute MUST be set to **true**.

3.1.5.3 Generating a Response to a SERVICE Request

SIP responses MUST have the following SIP fields set as specified:

- The **Content-Type** header field MUST be set to "application/SOAP+xml".
- A SIP response MUST contain a body unless the SIP response code indicates a failure. In the case of failure, the body SHOULD be present.
- In case of success, the body MUST contain the search results.

An example of a DS search response is found in section 4.1.

3.1.5.4 Error Responses

If all XML parsing failures:

- **SOAP fault code: Client.BadCall**
- **SIP return code:** 400

If there are Active Directory connectivity issues or when the server is busy:

- **SOAP fault code: Server.Unavailable.Busy**
- **SIP return code:** 503

If there are authorization-related error codes from the search:

- **SOAP fault code: Client.NotAuthorized.UserSearch**
- **SIP return code: 403 Forbidden**

For all other error results:

- **SOAP fault code: Server.UnknownError**
- **SIP return code:** 500

An example of a fault response to a DS user search is found in section 4.1. The fault response schema is specified in [SOAP1.1-Envelope].

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Enhanced Presence Publication Details

3.2.1 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.

An enhanced presence server maintains the following elements of state for each self subscriber:

SelfSubscriptionDialogEntry: Information about an individual self subscription dialog. This information includes the subscriber's URI and the subscribed URI. This entry also holds the self subscription scope, which MUST be any combination of the literal strings "categories", "containers", "subscribers" and "delegates". The **SelfSubscriptionDialogEntry** is a SIP SUBSCRIBE request, as defined in [RFC3265] section 3.1.

An enhanced presence server maintains the following elements of state for each category subscriber:

CategorySubscriptionDialogEntry: Each category subscriber is represented at the server by a **CategorySubscriptionDialogEntry** that has a list of publishers that the category subscriber is currently subscribed to. This information includes the subscriber's URI. Each category that is subscribed to has an associated container ID that corresponds to the publisher's container, to which the subscriber has been resolved for this category. This container ID is invalid if the subscriber did not resolve to any container. The **CategorySubscriptionDialogEntry** element also contains the state required to process a SIP SUBSCRIBE request, as specified in [RFC3265] section 3.1.

An enhanced presence server maintains the following elements of state for each publisher in addition to the publisher URI:

ContainerSet: A set of **ContainerEntry**, one **ContainerEntry** for each container that has an published member or a published **category** instance. Entries are keyed on the ID of the container.

ContainerEntry: Information about an individual container. This information includes the container ID. The entry also holds a set of the container's members and a set of categories that have one or more published instances in that container.

ContainerMemberSet: A set of entries, one for each member that belongs to the container. Entries are keyed on the container ID, the member type, and the member value. The set has an associated version number that is used to synchronize updates from multiple endpoints of the same publisher.

ContainerMemberEntry: Information about a member that belongs to the container. This information includes the container ID, the member type, which MUST be "domain", "everyone", "federated", "publicCloud", "sameEnterprise", or "user", and an optional member value. Members of type "domain" MUST have a value that specifies a domain name, and members of type "user" MUST have a value that specifies a **user-URI**. The member value MUST be omitted for all other member types.

ContainerCategorySet: A set of entries, one for each **category** that has a published instance in that container. Entries are keyed on the container ID and **category** name.

ContainerCategoryEntry: Information about an individual category belonging to the container. This information includes the container ID and the **category** name. The entry also holds a set of **category** instances and the related publisher URI. This set of **category** instances is built from published

instances of this **category** to the container. Each instance has an associated instance number, version number, an expiry type, which MUST be "static", "endpoint", "user", or "time", an optional expiry time, the ID of the publishing endpoint ID, and the published XML data. As always, the version number is used to synchronize updates from multiple endpoints of the same publisher. The version number allows multiple endpoints of the same publisher to synchronize their publications and allows self subscribers to determine whether to accept a notification. The **name** attribute of the **category** element in a categories document is compared to a list of the names of agreed-upon schemas.<31> The agreed-upon schema with an exact matching name is used to interpret the body of the element.

3.2.2 Timers

An enhanced presence server maintains the following timers for each publisher:

Publication Cleanup Timer: This is a periodic timer used to clean up stale time-bounded **category** instances for the publisher.<32>

Server Publication Timer: This is a periodic timer used to publish **category** instances into containers for each publisher, as specified in section 3.2.6.<33>

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

When a single or batched publication request is received from a Publisher, the UAS processes individual publication requests as specified in section 3.2.5.1.2.

3.2.5 Message Processing Events and Sequencing Rules

Except as specified in this section, the rules for message processing are as specified in [RFC3261] and [RFC3265].

3.2.5.1 Receiving a Batch Publish Request

Enhanced presence uses a **publish** request to publish **category** instances. Publishers can batch together publications that add, modify, and remove multiple **category** instances in a single request to the server. The **category** instances can be from different categories and can belong to different containers. Each publication in a batch publish request is processed as specified in section 3.2.5.1.2.

3.2.5.1.1 Indicating Support for Enhanced Presence

This protocol defines a new option tag to indicate support for SIP extensions for presence. The SIP protocol client MUST insert the Supported header with the following option tag in its REGISTER requests as part of the sign-in sequence:

```
Supported: msrtc-event-categories
```

3.2.5.1.2 Processing Publications

This section describes the Processing Publications procedure, as illustrated in section 4.2.1 where Bob@contoso.com is publishing an instance of the **note category** to containers 200, 300, and 400.

As specified in section 2.2.2.2.1, a **publish** request is identified by a SERVICE request with a Content-Type header field value of "application/msrtc-category-publish+xml". Upon receipt of such a message,

the server MUST verify that the **To-URI**, the **From-URI** in the SERVICE header, and the URI in the **publications** element are all identical. This URI is then used to look up the publisher whose **category** instances are to be modified.

For each publication element in the request, the server MUST first validate the different attributes to ensure they conform to the correct XML schema and meet the constraints specified in section 2.2.2.2.1. In particular, the expiry time MUST be included if the expiry type is set to "time". The **categoryName** attribute of **publication** element in a **publish** request is compared to a list of the names of agreed-upon schemas. The agreed-upon schema with an exactly matching name is used to interpret the body of the element.

If the value of **expireType** attribute is "endpoint", the client MUST register the endpoint first, as specified in [MS-SIPREGE] section 3.1.1.5.1 and section 3.1.2.5.1. The server MUST NOT allow publications for a category to go to a container whose name has been marked as read-only. The server SHOULD also limit publications to registered category names. The implementation can allow an administrator to register category names by some means outside the scope of this specification. In that case, the server SHOULD fail any publish request that includes a publication for an unregistered category name.

The server then looks up the publisher's **ContainerSet** for the **ContainerEntry** that has the specified container ID, creating the entry if one is not found. Next, the server looks up the **ContainerCategorySet** of this **ContainerEntry** for the **ContainerCategoryEntry** that has the specified category name, again creating the entry if one is not found. The rest of the description in this section refers to this **ContainerCategoryEntry**.

The server then validates the version number of the **category** instance. This validation is also referred to as a version check, which is performed as follows:

1. The server finds the **category** instance in the **ContainerCategoryEntry** that has the instance number and publisher URI specified in the **publication** element.

2. The version number associated with this **category** instance MUST equal the version specified in the **publication** element.

3. If there is no such **category** instance, the version specified in the **publication** element MUST be zero.

4. If a version check is successful, the action MUST result in the version number being incremented.

The server MUST set the **publishTime** of the **category** instance to the current time in UTC.

The server can now begin performing the action implicit in the **publication** element.

If the publication element specifies an expiry time of zero, the server deletes the existing **category** instance, if any, after removing it from the **ContainerCategorySet**. If the **category** instance does not exist, the server moves on to the next publication element. Otherwise, if the publication element specifies a nonzero expiry time, the server updates the existing **category** instance according to the specified values if any. If there is no existing **category** instance, the server creates one with the specified values and inserts it into the **ContainerCategorySet**.

The server SHOULD limit the size of data that a publisher can include in a **publication** element. The server SHOULD also limit the total size of data a publisher can have for a category, across all instances of the **category** in the different containers. The implementation can permit an administrator to configure these limits based on the needs of the users in a **deployment** by some means outside the scope of this specification. The server SHOULD fail any **publish** request that causes these limits to be exceeded.

When the **category** instance has been successfully created, removed, or updated; the server updates the version number associated with it and moves on to the next **publication** element.

The server MUST NOT commit any change until all **publication** elements in the **publish** request have been processed, cancelling pending changes if any operation fails. If all operations are successful, the server sends a 200 OK response with an embedded **categories** document, as specified in section 2.2.2.2.2. For each **category**, **publishTime** MUST be generated by the server according to when the **category** instance was published, and **version** MUST specify the current version number at the server.

An example of the 200 OK response is found in section 4.2.1 for the **publish** request in the example in section 4.2.1.

The 200 OK response MUST have a **Content-Type** header field value of "application/vnd-microsoft-roaming-self+xml" and MUST embed a **categories** document. Each **category** element of the **categories** document MUST refer to a **category** instance that is being notified. The **category** element MUST specify all attributes that have values. Although the last publish time and version number are optional values for most categories documents, the categories document in a 200 OK response MUST include the updated values of these attributes. The body of the element MUST contain the published XML data.

The set of **category** instances being notified MUST include the complete set of **category** instances from each **ContainerCategoryEntry** that has been modified because of the **publish** request.

3.2.5.2 Generating NOTIFY for Self Subscribers

Continuing from the example in section 4.2.1, the following message illustrates a BENOTIFY generated by the server on an active self subscription dialog.

[MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

An example of a BENOTIFY message generated by a server is found in section 4.4.3

After a publisher's **publish** request has been successfully processed, the server MUST generate a NOTIFY or BENOTIFY request for each corresponding **SelfSubscriptionDialogEntry**. A corresponding **SelfSubscriptionDialogEntry** contains a subscribed URI that is equal to the URI of the publisher, and its scope MUST include "categories". The NOTIFY or BENOTIFY request MUST be generated even for the endpoint that sent the **publish** request, as long as that endpoint has a corresponding **SelfSubscriptionDialogEntry**. The NOTIFY or BENOTIFY request is sent to every registered endpoint that has a corresponding **SelfSubscriptionDialogEntry**. A registered endpoint corresponds to a **SelfSubscriptionDialogEntry** when the SIP URI component of an active binding matches the subscriber's URI and the subscribed URI of the **SelfSubscriptionDialogEntry.** The creation and maintenance of an endpoint binding is specified in [MS-SIPREGE] section 3.1.2.5.1 and [RFC3261] section 10.

As illustrated in section 2.2.2.3.3, the generated NOTIFY or BENOTIFY request MUST have a Content-Type header field value of "application/vnd-microsoft-roaming-self+xml" and MUST embed a **categories** document. Each **category** element of the **categories** document MUST refer to a **category** instance being notified. The **category** element MUST specify all attributes, including the updated values of the last publish time and version number. The body of the element MUST contain the published XML data.

The set of **category** instances being notified MUST include the complete set of **category** instances from each **ContainerCategoryEntry** that has been modified because of the **publish** request.

3.2.5.3 Generating NOTIFY Requests for Category Subscribers

Modifications to the set of published **category** instances can potentially alter the container from which **category** instances are made visible to a category subscriber. Hence, the final step in processing a **publish** request involves reevaluating active category subscribers and, if required, generating NOTIFY

or BENOTIFY requests on their category subscription dialogs. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

Each category subscriber is represented at the server by a **CategorySubscriptionDialogEntry** that has a list of publishers that the category subscriber is currently subscribed to. The server reexamines every **CategorySubscriptionDialogEntry** that includes, in its publisher list, an entry for the SIP protocol client that made the **publish** request. The publisher entry has a list of subscribed-to categories. For each subscribed-to category in this list, the server reevaluates the **ContainerCategoryEntry** to which the category subscriber now resolves.

To resolve the **ContainerCategoryEntry** for a given publisher, category subscriber, and category combination, the server MUST consider only those **ContainerEntry** elements of the publisher that contain a **ContainerCategoryEntry** for the specified category. From these, the server MUST select the first, if any, **ContainerEntry** returned by the following rules, applied in order:

1. Select the highest numbered **ContainerEntry** that includes a **ContainerMemberEntry** of type "user" and a **user-URI** that equals the category subscriber's URI.

2. Select the highest numbered **ContainerEntry** that includes a **ContainerMemberEntry** of type "domain" and a domain name that equals the category subscriber's domain.

3. If the category subscriber is a same-enterprise user, select the highest numbered **ContainerEntry** that includes a **ContainerMemberEntry** of type "sameEnterprise".

4. If the category subscriber is a federated user, select the highest numbered **ContainerEntry** that includes a **ContainerMemberEntry** of type "federated".

5. If the category subscriber is a public cloud user, select the highest numbered **ContainerEntry** that includes a **ContainerMemberEntry** of type "publicCloud".

6. Select the **ContainerEntry** for the default container ID, if present.

The server MUST treat the specified category subscriber as resolved to the **ContainerCategoryEntry** from the selected **ContainerEntry**. If no **ContainerEntry** was selected, the server MUST treat the category subscriber as resolved to a null **ContainerCategoryEntry**.

If the reevaluated **ContainerCategoryEntry** represents a different container from that currently associated with the subscribed-to category, or if the set of published **category** instances for this **ContainerCategoryEntry** has changed in any way because of the **publish** request, the server MUST notify the category subscriber. To do so, the server adds the complete set of **category** instances from the **ContainerCategoryEntry** to the collection of **category** instances to be notified. If the **ContainerCategoryEntry** is null, a **category** instance with an empty data value is added to the collection instead. Additionally, if required, the server updates the container associated with the subscribed-to category.

After all subscribed-to categories have been reevaluated, if the collection of **category** instances to be notified is non-null, the server generates a NOTIFY or BENOTIFY request for the **CategorySubscriptionDialogEntry**. As illustrated in section 2.2.2.4.3, the generated NOTIFY or BENOTIFY request MUST have a **Content-Type** header field value of "application/msrtc-event-categories+xml" and MUST embed a **categories** document. Each **category** element of the **categories** document MUST refer to a **category** instance being notified. The **category** element MUST specify the **category** name, the instance number, and the last publish time. There MUST NOT be any specification of the instance's container ID, version number, expiry type, endpoint ID, or expiry time. The body of the element MUST contain the published XML data.

3.2.5.4 Error Responses

The server MUST send back a 400 Bad Request response if the request is missing an XML body.

The server MUST send back a 400 Bad Request response if the XML body is not a valid **http://schemas.microsoft.com/2006/09/sip/rich-presence** document.

The server MUST send back a 403 Forbidden response if the **From-URI** and the **To-URI** are not identical.

The server MUST send back a 400 Bad Request response if the To-URI and the URI in the publications element are not identical.

The server MUST send back a 400 Bad Request response if a time-bounded publication does not specify an expiry time.

The server MUST send back a 403 Forbidden response if a publication is attempted for a **category** to a container where the **category** name has been marked read-only.

The server MUST send back a 403 Forbidden response if a publication failed because it attempted to publish to an unregistered **category** name.

The server MUST send back a 409 Conflict response if the version check fails. The server MUST fail all publications in the request, even if only one has a version conflict. The format of this message is specified in section 2.2.2.2.3.

The server MUST send back a 400 Bad Request response if there are multiple publications for the same **category** instance in a container.

The server SHOULD send back a 413 Request Entity Too Large response if the request failed because the size of data include in a publication element exceeds a configured limit.

The server SHOULD send back a 413 Request Entity Too Large response if the request failed because it caused a **category** quota to be exceeded.

The server MUST return 488 if the value of **expireType** is "endpoint" but endpoint is not registered as specified as [MS-SIPREGE] section 3.1.2.5.1.

3.2.5.5 Endpoint Deregistration or Expiration

When a registered endpoint deregisters or its registration expires, as specified in [MS-SIPREGE] section 3.2, the enhanced presence server is notified to allow it to clean up any stale publications. The server uses the user URI from the deregistered endpoint to look up the publisher whose endpoint-bounded **category** instances MUST now be deleted.

For each **ContainerEntry** in the **ContainerSet** of this publisher, the server processes the **ContainerCategorySet**. For each **ContainerCategoryEntry** in this set, the server looks up **category** instances that have an expiry type of "endpoint" and a publishing endpoint ID that equals that of the expired endpoint. The value of **endpointId** is the UUID in the **+sip.instance** parameter value from the **Contact** header field, according to the syntax described in [MS-SIPRE] section 2.2.5. The server deletes these **category** instances after removing them from the **ContainerCategorySet**.

After all **ContainerEntry** elements have been similarly processed, the server MUST inform all self subscribers whose URI equals the URI of the publisher and all category subscribers that are affected by the changes. This involves generating NOTIFY or BENOTIFY requests for self subscribers whose URI equals the URI of the publisher, as specified in section 3.2.5.2, and generating NOTIFY or BENOTIFY requests for affected category subscribers, as specified in section 3.2.5.3. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

3.2.5.5.1 User Deregistration

When the last endpoint of a user deregisters, the enhanced presence server is notified to allow it to clean up any stale publications. The server uses the user URI from the deregistered endpoint to look up the publisher whose user-bounded **category** instances have expired.

For each **ContainerEntry** in the **ContainerSet** of this publisher, the server processes the **ContainerCategorySet**. For each **ContainerCategoryEntry** in this set, the server looks up **category** instances that have an expiry type of "user" as well as the **category** instances that have an expiry time of "endpoint" and a publishing endpoint ID that equals that of the expired endpoint. The value of **endpointId** is the UUID in the **+sip.instance** parameter value from the **Contact** header field according to the syntax described in [MS-SIPRE] section 2.2.5. The server deletes these **category** instances after removing them from the **ContainerCategorySet**.

After all **ContainerEntry** elements have been similarly processed, the server MUST inform subscribers of the changes. This involves generating NOTIFY or BENOTIFY requests for affected category subscribers, as specified in section 3.2.5.3. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

3.2.5.6 Aggregation

Each operation that causes a change in the **state** and **services category** instances in containers 2 and 3 triggers an aggregation, as specified in section 3.8. An example of such an operation is an endpoint that publishes a new presence state or expiration of a device **category** when the endpoint deregisters.

3.2.5.7 Publication of mwi category

The unified messaging server notifies the messaging server whenever the voice mail information changes for the user.<34> The unified messaging server also notifies the messaging server every 12 hours even if voice mail information is not changed for the user. When there is a notification from exchange unified messaging server for the user, the messaging server MUST extract the voice mail information from the notify and publish mwi category for that user using the format specified in section 2.2.2.7.11.

3.2.5.8 Publication of linkedPICContacts category

The SIP protocol client extracts the linked identities node from the NOTIFY response for a presence request for a presentity in the contact list and publish the identities as a linkedPicContactEntry in this category. The identities in the NOTIFY response are in the format that is documented in section 2.2.2.8.2. The information in this category is used by the client to keep track of all the URLs that should be given the same permission level with respect to incoming SIP Invites. All outgoing SIP Invites should be addressed to the URL in the primary-Identity attribute. The client must publish the category using the format specified in section 2.2.2.7.13

3.2.6 Timer Events

3.2.6.1 Publication Cleanup Timer

When a publisher's periodic publication cleanup timer fires, the server expires any stale time-bounded **category** instances for the publisher. For each **ContainerEntry** in the **ContainerSet** of this publisher, the server processes the **ContainerCategorySet**. For each **ContainerCategoryEntry** in this set, the server looks up **category** instances that have an expiry type of "time" and an expiry time that is in the past. The server deletes these **category** instances after removing them from the **ContainerCategorySet**.

After all **ContainerEntry** elements have been similarly processed, the server MUST inform all self subscribers whose URI equals the URI of the publisher and all category subscribers that are affected

by the changes. This involves generating NOTIFY or BENOTIFY requests for self subscribers whose URI equals the URI of the publisher, as specified in section 3.2.5.2, and generating NOTIFY or BENOTIFY requests for affected category subscribers, as specified in section 3.2.5.3, [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

3.2.6.2 Server Publication Timer

When a publisher's periodic publication timer expires, the server MUST publish certain **category** instances into various containers for each publisher, as specified in the following table. The server can acquire a user's information from a DS to publish **contactCard** and **userProperties category** instances. The publisher SHOULD NOT add any members in the container where the **userProperties category** instance is published to avoid subscriptions on this category from subscribers.

The following table lists the publication of **category** instances upon expiration of a publisher's periodic publication timer. The set of containers referenced are pre-defined. A container in the **ContainerSet** is created as specified in section 3.2.5.1.2 and section 3.5.5.2.

Category name	instance attribute	expireType attribute	Containers for the publication	Read only	Elements and attributes to publish
contactCard (See section 2.2.2.7.5 for the format)	0	static	0,100, 200, 300, 400, 32000	No.	**displayName** and **email** elements in the **identity** element. The server can get these values from corporate (Active) directory.
legacyInterop (See section 2.2.2.7.6 for the format)	0	static	32000		**availability** attribute in **legacyInterop** element. Value of **availability** MUST be hardcoded to offline class (18500).
userProperties (See section 2.2.2.7.4 for the format)	0	static	1	Yes for container 1.	Subset or full set of elements in **userProperties** element.

userProperties in container 1 MUST be registered as a read-only **category** to avoid the SIP protocol client overwriting this **category** instance.

The server SHOULD optimize by not publishing anything if the new information to be published is the same as existing information in a given container.

The examples in section 4.13 illustrate the kinds of publication performed by the server.

3.2.7 Other Local Events

None.

3.3 Enhanced Presence Self Subscription Details

For overview details about self subscriptions, see section 1.3.1.5.

3.3.1 Abstract Data Model

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

The abstract data model for enhanced presence self subscription is covered in section 3.2.1.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

When a self SUBSCRIBE request is received from a UAC, the UAS processes the self SUBSCRIBE request as specified in section 3.3.5.1.

3.3.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are specified as in [RFC3261] and [RFC3265].

3.3.5.1 Processing a Self SUBSCRIBE Request

SIP protocol clients subscribe to self data after sending the REGISTER request as part of the sign-in sequencing rule. Self SUBSCRIBE is used to retrieve roaming self-state, which is containers, publications, subscribers and delegates, from the server. It is also used as the synchronization mechanism between multiple endpoints. That is, when another endpoint of the same user causes any self state, which is one of containers, subscribers, publications, or delegates, to change, all self subscribers are notified. An example is an endpoint that changes presence state from "online" to "busy." In such a case, all of the user's endpoints are expected to change state to "busy." State change, or category publication, causes all endpoints to be notified and to become synchronized as a result. Similarly, when an endpoint adds a container member, all other endpoints are notified so that the change is reflected in the user interface.

In the sample in section 4.4.1, the user subscribes to containers, categories, subscribers and delegates. It is possible to subscribe to any combination of the four roaming types, which are categories, containers, subscribers and delegates.<35> SIP protocol clients ordinarily subscribe to all four types.

The content of the SUBSCRIBE request is described by the XML schema that can be found in section 6. The schema allows the SIP protocol client to define, as well as subsequently modify, the list of roaming types, or scope, that are subscribed to. The roaming type list in each self SUBSCRIBE request that is sent using the same SUBSCRIBE dialog overwrites the roaming type list stored on the server. If

an initial self SUBSCRIBE request lists "categories" and a subsequent request lists "subscribers", the resulting SUBSCRIBE dialog scope on the server will be "subscribers".

Each user endpoint creates a single self SUBSCRIBE dialog with the server with all four roaming types. The server SHOULD restrict the number of self subscription dialogs to one by allowing subsequent enhanced presence self SUBSCRIBE dialog creation requests from the same user endpoint while terminating the existing dialog by sending a NOTIFY with **Expires** set to zero.

After receiving a request to create a self SUBSCRIBE dialog, the server parses the "roamingList" to determine the scope defined by this request. Based on this scope, which is a combination of "containers", "categories", "subscribers", and "delegates", the server queries its internal state to gather the information necessary to generate the XML documents defined in section 2.2.2.3. If this is a dialog creation request, a new self subscription dialog, **SelfSubscriptionDialogEntry**, is created. Otherwise, the existing dialog is updated. The server MUST return all data defined in the scope, even in the case of a subscription refresh request.

If the **UAC** is in **survivable mode**, as specified in [MS-SIPREGE] section 3.1.1.5.1, the server MAY not create a new self subscription dialog, **SelfSubscriptionDialogEntry**, even if this is a dialog creation request.<36>

In the case of a polling self subscribe, the server MUST NOT store the **SelfSubscriptionDialogEntry**.

If the self subscription operation is successful, the server MUST return a 200 OK response. A typical SIP response to a self subscription appears in section 4.4.2. This is a piggybacked 200 OK response to a SUBSCRIBE request.

3.3.5.2 Generating a NOTIFY Response to a Self SUBSCRIBE Request

Each operation that causes a change in a user's publications, containers, subscribers and delegates, such as an endpoint deleting a publication, triggers a NOTIFY response for the self subscription dialog. During generation of the NOTIFY response, the server goes over all existing **SelfSubscriptionDialogEntry** elements for the user, and if the operation changes the data within the scope of a self subscription dialog, it sends a NOTIFY response to the owner endpoint of the dialog. A self SUBSCRIBE NOTIFY response is generated even for the endpoint that performs the operation that triggers the NOTIFY response. These operations can also be server actions, such as expiration of publications.

If an endpoint is registered to, and also has a self subscription dialog with, the same server, NOTIFY responses to be sent on the self subscription MAY use the registration route set instead of the self subscription dialog route set for routing NOTIFY responses. This is done because registration refresh is more frequent than subscription refresh, which makes the registration route more reliable.

[MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

Section 4.4.3 contains a sample BENOTIFY triggered as a result of publication.

3.3.5.3 Error Responses

The server MUST send back a 400 response when an incoming XML document is not a valid **http://schemas.microsoft.com/2006/09/sip/roaming-self**, including the **http://schemas.microsoft.com/2007/09/sip/roaming-self-ex** additions in section 6.6, document.

The server MUST send back a 404 Not Found response if the **From-URI** and the **To-URI** are not identical and the **To-URI** is not a configured user. The server MUST send back a 400 Bad Request response if the **From-URI** and the **To-URI** are not identical and the **To-URI** is a configured user. The behavior when **Request-URI** is not identical to the **From-URI** and the **To-URI** is not specified.

The server MUST send back a 400 response if the request is missing an XML body.

The server SHOULD send back a 401 Unauthorized response when the endpoint used in a self subscription creation or update request is no longer registered.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

3.4 Enhanced Presence Category Subscription Details

For overview details about category subscriptions, see section 1.3.1.4.

3.4.1 Abstract Data Model

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

The abstract data model for enhanced presence category subscription is covered in section 3.2.1.

3.4.2 Timers

None.

3.4.3 Initialization

None.

3.4.4 Higher-Layer Triggered Events

When a single or batched category SUBSCRIBE request is received from a UAC, the **UAS** processes individual SUBSCRIBE requests as specified in section 3.4.5.1.

3.4.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261] and [RFC3265].

3.4.5.1 Processing Single and Batched Category SUBSCRIBE Requests

Enhanced presence uses **application/msrtc-adrl-categorylist+xml** document format for subscribing to a list of categories of a list of publishers. This request is similar to the batched SUBSCRIBE request, which is specified in [MS-SIP] section 3.3.4.1; however, the underlying presence model used here is encapsulated by enhanced presence categories as described in section 2.2.2.7 and not **msrtc.pidf**. Enhanced presence uses categories to define various presence-related data, such as calendar information and state. SIP protocol clients can subscribe to a list of these categories by sending a batched SUBSCRIBE request to the enhanced presence server. This SHOULD be done after the SIP protocol client has signed in to the server by sending a REGISTER request. The batched

SUBSCRIBE request is a SUBSCRIBE request in which the body of the request contains the contact URIs and categories of interest.

In the example in section 4.5, alice@contoso.com subscribes to the **contactCard**, **note**, **services**, **calendarData** and **state** categories of john@contoso.com.

3.4.5.1.1 Single vs. Batch Requests

The server is capable of handling both batch and single SUBSCRIBE requests. The main difference is that a single category SUBSCRIBE request MUST have only one resource in its resource list. A single category SUBSCRIBE request can still include multiple categories, but only for a single resource. Single category SUBSCRIBE requests are used to subscribe to users outside the deployment because the **To-URI** and the **From-URI** can be different.

When an enhanced presence category SUBSCRIBE request is received, the server inspects its **Require** header to determine whether it contains both **adhoclist** and **categoryList** option tags. If so, the request is considered to be a batched category subscribe. Otherwise, the request is considered to be a single category subscribe. The XML schemas for both persistent and polling requests and their responses are identical.

Other differences are as follows:

- A batch category SUBSCRIBE MUST have the following **Require** header:

```
Require: adhoclist, categoryList
```

- Both batch and single category SUBSCRIBE requests MUST have the following **Accept** header. If more documents are listed in the **Accept** header, the server MUST ignore them.

```
Accept: application/msrtc-event-categories+xml, application/rlmi+xml, multipart/related
```

- A batch category SUBSCRIBE request MUST have the same To-URI and From-URI. There can be a server GRUU in the URI of the request, as specified in [MS-SIPRE] section 2.2.3. A single category SUBSCRIBE request can have different URIs in the **From-URI**, the **To-URI**, and in the **Request-URI**. The server MUST route this request based on these headers. The **To-URI** and the **URI** listed in the resource list MUST be the same for a single category SUBSCRIBE request.
- The body of a single category SUBSCRIBE MUST NOT have more than one entry in its resource list (**adhocList**).
- The **name** attribute of the **action** element MUST NOT be set to "unsubscribe" for a single category subscribe.
- There MUST be only one **action** element for a single category SUBSCRIBE request.

Other than as noted in the preceding list, the processing of batched and single category subscribes is identical.

3.4.5.1.2 Polling vs. Persistent Category Subscriptions

There are two ways for a SIP protocol client to obtain another user's presence information. If the SIP protocol client needs only a one-time snapshot of the presence information, it SHOULD send a **Polling SUBSCRIBE** request, which means **Expires**: 0 with no tag in the **To** header field, to the server to do so. This approach can be useful when a user's presence state is needed only on a transient basis. The advantage of using a polling SUBSCRIBE request is that it does not consume many resources on the server.

If the SIP protocol client intends to get updates continually when presence information changes, it SHOULD subscribe to them by sending a (persistent) SUBSCRIBE request to the server. The SUBSCRIBE request creates a dialog, and as long as the dialog remains alive, the server MUST send presence updates in NOTIFY or BENOTIFY requests, as defined in [MS-SIP] section 3.5. This mode of operation can be useful for getting presence information about users in the SIP protocol client's contact list. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

When an enhanced presence category SUBSCRIBE request is received, the server inspects the **Expires** and **To** header fields. If the value in the **Expires** header field is zero, and there is no "tag" parameter in the **To** header field, the request is considered a polling category subscribe. Otherwise it is considered to be a persistent category subscribe. XML schemas for both request and response are identical.

The server MAY reject category subscriptions, both polling and persistent, with a 403 Forbidden response when the user named in the **To** header field is outside the deployment.

3.4.5.1.3 Processing Details

The content of the SUBSCRIBE request is described by the XML schema that can be found in section 6. The schema allows the SIP protocol client to define, as well as subsequently modify, a list of contacts, or publishers, and a list of categories per contact established by a previous SUBSCRIBE request in the same SIP SUBSCRIBE dialog. When one of the categories of one of these contacts changes, the server notifies the subscriber. The size of the list SHOULD be bounded by the maximum number of contacts per user setting on the server. The server SHOULD limit the maximum number of contacts that a user can have. The server SHOULD also limit the maximum number of categories that a user can subscribe to. The implementation SHOULD permit an administrator to configure these limits based on the needs of the users in a deployment, by some means outside the scope of this specification. The server SHOULD reject any batched SUBSCRIBE request that contain more contacts or categories than these limits.

The server can enforce a quota on the total or per context size of the context elements stored per publisher and return an error if this quota is exceeded.

Each user endpoint creates only one batched category SUBSCRIBE dialog with a server. The server can enforce this limit by allowing subsequent enhanced presence batched SUBSCRIBE dialog creation requests from the same user endpoint while terminating any existing ones by sending a NOTIFY with **Expires** with a value of zero.

A server (or server pool) can enforce a requirement that an endpoint be registered before the category SUBSCRIBE is accepted if the subscriber is located at the accepting server (or server pool). If the endpoint that sends the SUBSCRIBE request is not registered, the server SHOULD send back a 401 error response. For details, see section 3.4.5.3. If the server is implemented this way, operations that cause endpoints to be deregistered or cleaned up, such as endpoint expiration or deregistration, can clean up a subscription state that is associated with this endpoint as well.

The server MUST NOT allow category subscription requests to categories marked as "private". The server MUST ignore "private" categories in the **categoryList** body.

After receiving a batched category SUBSCRIBE dialog creation request, which means an **Expires** header with a value other than zero and an empty **tag** parameter in the **To** header field, the server parses the XML body. Initial parsing is done over the resource list to determine whether there are resources for which the server cannot handle subscriptions. These are called "rejected resources". Rejection cases are specified in section 3.4.5.1.3.1.

After parsing **resourceList**, the server determines which resources to reject.

For resources that are not in the rejection list:

- If the **action** is specified as "subscribe", the server parses the **categoryList** and creates the internal dialog state. Based on the data in the XML, a **CategorySubscriptionDialogEntry** is created with its publishers and categories members. The server then goes over the list of publishers and categories in the request and queries the internal state to determine which publications to return to the subscriber based on publisher access levels. The access level is set by the publishers or resources in the request using their containers. The server uses the logic described in section 1.3.1 to determine the container to use for each publisher or resource in the batch and also updates the **CategorySubscriptionDialogEntry** state to store the container numbers. The server then returns the publications from that container for the given category to generate the XML for the second part of the body in the response.

- If the **action** is specified as "unsubscribe", the server parses the **categoryList** and queries its internal state to find the **CategorySubscriptionDialogEntry**. If not found, a 200 OK is generated. If found, listed categories and resources are removed from the internal state and a 200 OK response is generated.

If the request received is a dialog refresh request, which means that the **Expires** header value is not equal to zero and the **tag** parameter in the **To** header field is equal to the **To** tag value of the dialog creating response sent from the server as specified in [RFC3265], the server queries its internal state to find the **CategorySubscriptionDialogEntry** of the subscribing user based on the subscriber's URI. If the **CategorySubscriptionDialogEntry** is not found, the request is rejected with a 481 response. For details, see section 3.4.5.3. If the **CategorySubscriptionDialogEntry** is found, the server parses the **resourceList,** as explained previously, to determine which resources to reject. The server then parses the **categoryList**. If any of these resources is not found as part of the **CategorySubscriptionDialogEntry**, the internal state is updated with the publisher and all categories from **categoryList**. If the publisher is found, the server goes over all categories to ensure that all categories in **categoryList** exist as part of the internal state. The server then uses the logic described in section 1.3.1 to determine the container to use for each publisher or resource in the batch if the container number is missing from the internal state, and also updates the **CategorySubscriptionDialogEntry** state to store the container numbers. The server then returns the publications from that container for the given category to generate the XML for the second part of the body in the response.

As seen in the example in section 3.4.5.1, an optional **context** element can be added to a **resource** element in a batched SUBSCRIBE request. This element can be added to more than one resource. Adding this element causes the server to add the subscriber to the resource's subscriber list. The server then generates a self SUBSCRIBE NOTIFY request with a **subscribers** body, as specified in section 2.2.2.3.3. This NOTIFY request is sent to all self SUBSCRIBE dialogs of the resource, using **SelfSubscriptionDialogEntry** structure queries for this resource URI. In the preceding example, self subscription dialogs for alice@contoso.com receive this NOTIFY request. The server updates internal state with the content of the **context** element, display name (**displayName**) and source network (**type**) for the SUBSCRIBE request ("sameEnterprise", "federated", or "publicCloud") using an **ms-source-network** header that is received in the SUBSCRIBE request. **SubscriberEntry** is created as a result, as specified in section 3.6.1. For details, see [MS-SIPRE] section 3.10.1.4. If an incoming SUBSCRIBE request is not marked as "publicCloud" or "federated", the server treats it as if it were marked "sameEnterprise". In the preceding example, john@contoso.com included a **context** element in the **resource** element for alice@contoso.com to indicate the request that is to be added to her subscriber list. If John is not already in Alice's subscriber list, Alice receives a new NOTIFY request informing her about this new user in her subscriber list. Alice's SIP protocol client can then prompt her for acknowledgement. For details about the subscriber list, see section 3.6.

The server can restrict creation of more than one batched category SUBSCRIBE dialog by the same subscriber endpoint.

A batch SUBSCRIBE dialog can handle multiple category subscriptions for multiple collocated resources. Whenever possible, SIP protocol clients SHOULD associate new subscriptions with an existing dialog when there is a need to subscribe to new resources or new categories of existing resources. Subscribers SHOULD send an initial batch category SUBSCRIBE request to their server, using their **From-URI** and **To-URI** and create the rest of the subscription dialogs based on the

response that provides details about what to do for resources that are not located on that server or server pool. Similarly, when a NOTIFY request with the **Expires** header field set to zero is sent back to the SIP protocol client by a server to terminate an existing category subscription dialog, SIP protocol clients SHOULD include the resources that were present in the terminated dialog in the batched category SUBSCRIBE dialog that they maintain with the server on which they are registered. If one is not present, they SHOULD create a new category subscription dialog and include all resources in it.

If the batched subscription operation is successful, the server MUST return a 200 OK response. To view a typical 200 OK SIP response to a batched subscription, see section 4.5. It is a multi-part MIME body that contains both a rejection list and the category data for each contact that was subscribed to successfully.

The boundary string "fd6a7790014f493faeaf225c3a2eb3d9" in the referenced example delimits parts of the multi-part body. The usage of a boundary string in MIME is described in [RFC1341]. The first part is a list, expressed in XML format, that contains the parts of this batch subscription request that the server could not fulfill. This is called the rejection list. There can be various reasons why the server is not able to handle parts of a batch subscription request. In all these cases, the **instance** attribute MUST have the information specified in section 3.4.5.1.3.1. The **instance** attribute information provides the reason for the rejection. The SIP protocol clients can act on the response based upon the provided information.

The remaining parts are categories in **application/msrtc-event-categories+xml** format, as specified in section 2.2.2.1.

After the 200 OK response is sent back to the SIP protocol client, the server is responsible for storing the subscription state, for generating NOTIFY messages, and for sending them to all registered endpoints of the user. A NOTIFY message MUST be generated whenever there is a change in the data that is sent back to the subscriber in response to the subscription request.

The internal state mentioned in this section is transient in the case of polling category subscribes. Unlike **persistent SUBSCRIBEs**, after a 200 OK with the response body is sent, the server cleans up any state related to the polling SUBSCRIBE.

3.4.5.1.3.1 Batched Category SUBSCRIBE Request Rejection Scenarios

Resources can be rejected in any of the following cases:

- The user is not collocated with the subscriber on the same server resource.
- The user is from a different deployment. SIP protocol clients SHOULD send a new single category SUBSCRIBE request in this case by supplying the URI of the resource as the **To-URI**. When bob@fabrikam.com subscribes to alice@contoso.com, the server sends back the following:

```
<instance id="0" state="resubscribe" cid="alice@contoso.com"/>
```

- The user is from the same domain but is hosted on a different server or server pool. In this case, the server sends back a **poolFqdn** attribute, which supplies a URI that routes the request to a server or server pool that can accept a batch subscription for the user. The server uses a pool GRUU for routing. For details, see [MS-SIPRE] section 2.2.3. The following example shows the GRUU of the server that hosts alice@contoso.com:

```
sip:server.contoso.com@contoso.com;gruu;opaque=srvr:HomeServer:dL8cwxBrTuG8eC4-Q_GNGAAA.
```

Whenever possible, SIP protocol clients SHOULD associate new subscriptions with an existing dialog when there is a need to subscribe to new resources or new categories of existing resources.

- SIP protocol clients SHOULD send a new batch category SUBSCRIBE request to the server GRUU supplied in **poolFqdn** and associate all future requests for all resources on that server or server pool with the same dialog. In the following example, bob@contoso.com subscribes to alice@contoso.com, who is located on the server represented by the preceding GRUU:

```
<instance id="0" state="resubscribe" cid="alice@contoso.com"
poolFqdn="sip:server.contoso.com@contoso.com;gruu;opaque=srvr:HomeServer:dL8cwxBrTuG8eC4-
Q_GNGAAA"/>
```

Note that the **poolFqdn** attribute is server implementation-specific and MAY not be required for all implementations.

The user has exceeded subscription limits.

- The server SHOULD enforce a limit on incoming subscriptions per user. If a user exceeds this limit, new subscriptions SHOULD fail. An **instance** element SHOULD contain an **id** attribute with a value of zero, a **state** attribute with a value of "terminated", a **reason** attribute with a value of "quotaExceeded", a **statusCode** attribute with a value of "413", and a **cid** attribute with the value formatted as a resource URI. The **cid** attribute value MUST resolve to the user who has more than the permitted number of users subscribed.

The following example indicates that alice@contoso.com has more than the permitted number of users subscribed to her.

```
<instance id="0" state="terminated" reason="quotaExceeded" statusCode="413"
cid="alice@contoso.com"/>
```

Note that enforcing limits on incoming subscriptions per user is server implementation-specific and MAY not apply to all implementations.

3.4.5.2 Generating a NOTIFY Response to a Batched SUBSCRIBE Request

Each operation that causes a change in the category data that is visible to the subscriber, such as an endpoint deleting a publication from a container, generates a NOTIFY response to update the SIP protocol client state. As the NOTIFY message is generated, the server goes over all **CategorySubscriptionDialogEntry** instances that involve this publisher and sends a NOTIFY response to the subscribing endpoints of the subscriber with the new data. These operations can also be server actions, such as expiration of time-bounded publications.

The server MAY use the registration route set instead of the category subscription dialog route set for routing the NOTIFY message if an endpoint is registered to and also has a category subscription dialog with the same server. This is done because registration refresh is done more frequently than subscription refresh, which makes the registration route more reliable.

The **version** attribute MUST be 0 for the first NOTIFY and MAY increase by one or be identical in value to the previous NOTIFY for each subsequent NOTIFY sent within this dialog.

[MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

An example in section 4.4.3 shows a BENOTIFY that is triggered by a **calendarState** publication operation.

3.4.5.3 Error Responses

The server MUST send back a 400 response when an incoming XML document is not a valid **http://schemas.microsoft.com/2006/01/sip/batch-subscribe** document.

The server MUST send back a 400 response if the request is missing its XML body.

The server MUST send back a 403 response if the **From-URI** and the **To-URI** are not the same for a batch category subscribe.

If a resource is not found, the following occurs:

- For a batch category SUBSCRIBE request, the server MUST send back a "resubscribe", which was described previously, and a hint for this resource in the rejection list.
- For a single category SUBSCRIBE request, the server MUST send back a 404 response.

If the request is a single category subscribe, the following occurs:

1. The server MUST send back a 400 response if there is more than one resource in the **resourceList** for a single category subscribe.
2. The server MUST send back a 400 response if there is more than one **action** element in the XML body.
3. The server MUST send back a 400 response if the **action** is defined as "unsubscribe".
4. The server MUST send back a 400 response if the **To-URI** is different from the URI in the resource list.
5. The server SHOULD send back a 401 when the endpoint used in a category subscription creation or update request is located on this server or server pool but is no longer registered.

The other errors that can be returned in the response (481 and 488) are mentioned in sections 3.4.5.1.3 and 3.2.5.4.

The server SHOULD have settings that control which users are allowed interactions, such as subscribing to presence, with users from federated domains or public clouds. In that case, the server SHOULD send back 403 Forbidden responses to outgoing SUBSCRIBE requests destined to **federated** or **publicCloud** resources if the subscriber is not granted federation or public cloud interaction rights. Similarly, the server SHOULD send back 403 Forbidden responses to incoming SUBSCRIBE requests coming from federated users or public cloud users if these SUBSCRIBE target users are not granted federation or public cloud interaction rights.

3.4.6 Timer Events

None.

3.4.7 Other Local Events

None.

3.5 Enhanced Presence Container Management Details

For overview details about container management, see section 1.3.1.3.

3.5.1 Abstract Data Model

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

The abstract data model used in container management is defined in section 3.2.1.

3.5.2 Timers

None.

3.5.3 Initialization

None.

3.5.4 Higher-Layer Triggered Events

When a **setContainerMember** request is received from a UAC, the UAS processes individual setContainerMember requests as specified in section 3.5.5.1.

3.5.5 Message Processing Events and Sequencing Rules

Except as specified in this section, the rules for message processing are as specified in [RFC3261] and [RFC3265].

3.5.5.1 Processing a setContainerMember Request

Enhanced presence uses a **setContainerMember** request for modifications to container membership. Publishers can batch multiple member addition and removal operations on multiple containers in a single request to the server.

3.5.5.2 Processing Requests

In the example found in section 4.9, bob@contoso.com is adding john@contoso.com to container 32000.

As specified in section 2.2.2.5.2, a **setContainerMember** request is identified by a SERVICE request with a Content-Type header field value of "application/msrtc-setcontainermembers+xml". Upon receipt of such a message, the server MUST verify that the **To-URI** and the **From-URI** in the SERVICE header fields are all identical. This **URI** is then used to look up the publisher whose container membership is to be modified.

For each **container** element in the request, the server MUST first validate the different attributes to ensure they conform to the XML schema specified in section 2.2.2.5.2. In particular, the container ID MUST NOT refer to the default container, which is container zero. The server then looks up the publisher's **ContainerSet** for the **ContainerEntry** that has this ID, creating the entry if one is not found. If a new entry is created, it MUST have a **ContainerMemberSet** that is empty and whose associated version number is set to zero. The rest of the description in this section refers to the **ContainerMemberSet** of this **ContainerEntry**.

The next step in processing the container element is to validate the version. The specified version MUST equal the version number associated with the **ContainerMemberSet**. The server then starts performing the actions specified in the member elements contained in the **container** element under consideration.

The server MUST perform the actions only if the version number specified in the request matches the current version, or if the specified version number is zero and this is the first action to the container membership. If successful, the action MUST result in the version number being incremented.

For each member element, the server validates the specified **type**, **value**, and **action**. The **type** MUST be "user", "domain", "sameEnterprise", "federated", "publicCloud", or "everyone". The value

MUST conform to domain name space rules as specified in [RFC1034] section 3.1 and resolve to an IP address if the type is "domain". If the type is "user", it MUST conform to SIP URI standards as specified in [RFC3261] section 19.1. The value MUST NOT be present for any other type. The action, if specified, MUST be either "add" or "delete". If the action is not specified, it is assumed to be "add".

For each member element with an action of "add", the server creates a new **ContainerMemberEntry** with the specified type and value and inserts it into the **ContainerMemberSet**. For each member with an action of "delete", the server removes the existing **ContainerMemberEntry** with the specified type and value from the **ContainerMemberEntry** that already exists and then deletes it. A request to "add" a **ContainerMemberEntry** that already exists or to "delete" a **ContainerMemberEntry** that does not exist is considered invalid and MUST be silently ignored.

The server SHOULD limit the total number of **ContainerMemberEntry** elements that can exist for a publisher across all of its containers. The implementation SHOULD permit an administrator to configure these limits based on the needs of the users in a deployment by some means outside the scope of this specification. The server SHOULD fail any **setContainerMember** request that causes this limit to be exceeded.

When all member elements for the container element have been successfully processed, the server updates the version number associated with the **ContainerMemberSet** and moves on to the next container element.

The server MUST NOT commit any change until all member elements for all container elements in the **setContainerMember** request have been processed, cancelling the pending changes if any operation fails. If all operations are successful, the server sends a 200 OK response, as specified in [RFC3261] section 13.2.2.4.

3.5.5.3 Generating NOTIFY for Self Subscribers

Continuing from the preceding example, the BENOTIFY message found in section 4.9.1 illustrates a BENOTIFY generated by the server on an active self subscription dialog. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

After a publisher's **setContainerMember** request has been successfully processed, the server MUST generate a NOTIFY or BENOTIFY request for each corresponding **SelfSubscriptionDialogEntry**. For a **SelfSubscriptionDialogEntry** to correspond, its subscriber URI MUST be identical to the publisher's URI and its scope MUST include "containers". The NOTIFY or BENOTIFY request MUST be generated even for the endpoint that sent the **setContainerMember** request, as long as that endpoint has a corresponding **SelfSubscriptionDialogEntry**.

As illustrated in section 2.2.2.3.3, the generated NOTIFY or BENOTIFY request MUST have a Content-Type header field value of "application/vnd-microsoft-roaming-self+xml" and MUST embed a **containers** document. Each **container** element of the **containers** document MUST refer, by the container ID, to a **ContainerMemberSet** that has been modified and MUST specify its updated version number. The member elements wrapped by the **container** element MUST together represent the complete new membership of the **ContainerMemberSet**. If the **ContainerMemberSet** is now empty, the **container** element MUST have an empty body.

3.5.5.4 Generating NOTIFY for Category Subscribers

Modifications in container memberships can alter the container from which **category** instances are made visible to a category subscriber. Hence, the final step in processing a **setContainerMember** request is to reevaluate active category subscribers and, if required, to generate NOTIFY or BENOTIFY requests on their category subscription dialogs as specified in section 3.2.5.3. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

3.5.5.5 Error Responses

The server MUST send back a 400 Bad Request response if the request is missing its XML body.

The server MUST send back a 400 Bad Request response if the XML body is not a valid **http://schemas.microsoft.com/2006/09/sip/container-management** document.

The server SHOULD send back a 403 Forbidden response if the **From-URI** and the **To-URI** are not identical<37>.

The server MUST send back a 409 Conflict response if the version check fails. The format of this message is specified in section 2.2.2.10.4.

The server MUST send back a 400 Bad Request response if there are multiple **container** elements for the same ID.

The server MUST send back a 413 Request Entity Too Large response if the request failed because it caused the container quota to be exceeded.

3.5.6 Timer Events

None.

3.5.7 Other Local Events

None.

3.6 Enhanced Presence Subscriber Management Details

For overview details about subscriber list management, see section 1.3.1.4.

3.6.1 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.

In addition to the abstract data model defined in section 3.2.1, the server keeps track of a list of subscribers per publisher.

SubscriberEntry: Information about a subscriber in publisher's subscriber list. This information includes the publisher's URI, subscriber's URI, subscriber's display name, acknowledgement bit, an optional source network type and an optional context member.

SubscriberSet: Each publisher has a **SubscriberSet** with one or more **SubscriberEntry** elements in it. A **SubscriberSet** object contains a set of entries and is keyed to the publisher URI.

3.6.2 Timers

None.

3.6.3 Initialization

None.

3.6.4 Higher-Layer Triggered Events

When a **setSubscriber** request is received from a UAC, the UAS processes individual **setSubscriber** requests as specified in section 3.6.5.1.

3.6.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are specified as in [RFC3261] and [RFC3265].

3.6.5.1 Processing a setSubscriber Request

As specified in section 2.2.2.6.2, a **setSubscriber** request is identified by a SERVICE request with a Content-Type header field value of "application/msrtc-presence-setsubscriber+xml". Upon receipt of such a message, the server MUST verify that the **To-URI**, the **From-URI**, and the **URI** in the SERVICE header are identical. This URI is then used to look up the publisher whose subscriber list is to be modified. The server then parses the XML to fetch the subscriber URI and looks up this URI in the **SubscriberSet**. If a **SubscriberEntry** for this subscriber URI is not found, the server returns a 400 error response. If a **SubscriberEntry** for this subscriber URI is found, the server updates the internal state according to the **acknowledged** attribute in the **subscriber** element and then sends a 200 OK. If the **acknowledged** value is "true", the server SHOULD remove the **SubscriberEntry** from the **SubscriberSet**. This is done to avoid an ever-growing **SubscriberSet** structure for all users; otherwise, **SubscriberSet**s grow over time as users' contact lists grow.

The removal of a **SubscriberEntry** when the **acknowledged** attribute is "true" is not possible in the case of MSRTC and PIDF subscriptions. The server MUST NOT remove the subscriber entries for these subscriptions, because MSRTC and PIDF subscriptions do not have a similar "context" concept. If the server were to remove the subscriber entries, Publishers would be notified every time they get subscribed by MSRTC/PIDF subscribers. For details, see section 3.7.5.3.

The server SHOULD enforce a size limit for the **SubscriberSet** list. The size limit is defined by an implementer. The server SHOULD return a 413 error response if the number of entries in the **SubscriberSet** list exceeds an enforced server limit.

If there is a non-empty **context** element in the category SUBSCRIBE request that is received from a subscriber, the server MUST insert the content of the non-empty **context** element in the **context** subelement of the **subscriber** element specified in section 2.2.2.6.1; otherwise, this element is omitted. For an example of a SUBSCRIBE request with a non-empty **context** element, see section 2.2.2.4.1.

The server forces a total size limit for the context elements stored per publisher. The size limit is defined by an implementer. The server can return a 413 error response.

The XML schema for the document can be found in section 6. A **setSubscribers** request MUST NOT support acknowledging multiple subscribers at once, as seen in the schema.

In the example found in section 4.8.1, Bob acknowledges alice@contoso.com.

3.6.5.2 Generating a Subscriber NOTIFY for Self Subscribers

Except as specified in the following section, the rules for message processing are specified as in [RFC3261] and [RFC3265].

After a publisher's **setSubscriber** request has been successfully processed, the server MUST generate a NOTIFY or BENOTIFY request for each corresponding **SelfSubscriptionDialogEntry**. For a **SelfSubscriptionDialogEntry** to correspond, its subscriber URI MUST be identical to the URI of the publisher, and its scope MUST include "subscribers". The NOTIFY or BENOTIFY request MUST be generated even for the endpoint that sent the **setSubscriber** request, as long as that endpoint has a

corresponding **SelfSubscriptionDialogEntry**. The NOTIFY or BENOTIFY request includes all entries in the **SubscriberSet** and not just the delta. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

Section 4.4.3 includes an example self subscription NOTIFY that is generated as a result of a **setSubscriber** request.

The following figure shows a typical subscriber list call flow. Alice adds Bob to her contact list and sends a category SUBSCRIBE request with "context" that triggers a self SUBSCRIBE NOTIFY for Bob, and then acknowledges Alice.

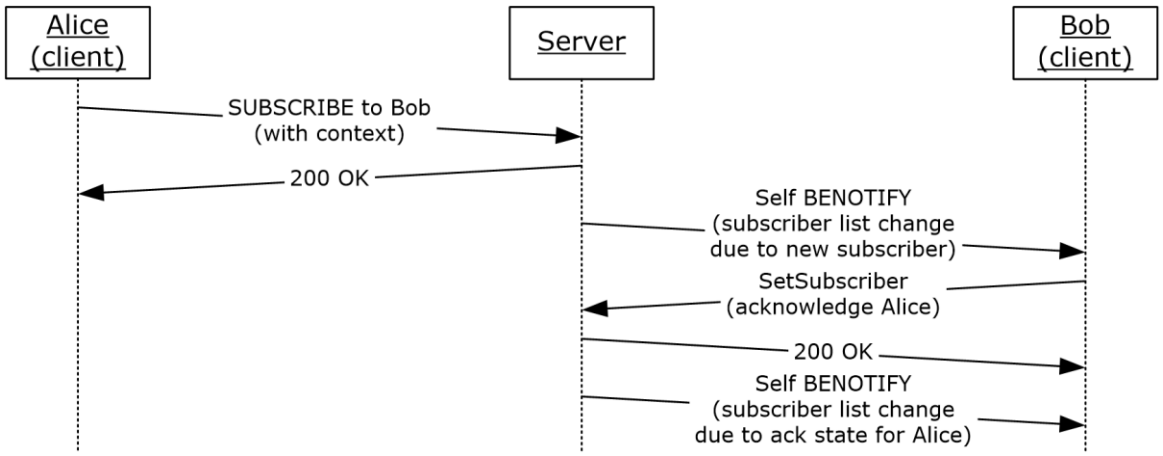

Figure 2: Sample call flow showing a new subscriber being added to subscriber list

Section 4.8 shows the actual requests and responses for the flow in the preceding figure.

3.6.5.3 Error Responses

The server MUST send back a 400 response when the incoming XML document is not a valid **http://schemas.microsoft.com/2006/09/sip/presence-subscribers** document.

The server MUST send back a 400 if the publisher sends a **setSubscriber** request with a subscriber URI that cannot be found in **SubscriberSet**.

The server SHOULD send back a 403 Forbidden response if the **From-URI** and the **To-URI** are not identical<38>.

The server MAY send back a 413 response if the total size for context elements in **SubscriberSet** exceeds an enforced server limit.

The server SHOULD send back a 413 response if the number of entries in **SubscriberSet** exceeds an enforced server limit.

3.6.6 Timer Events

None.

3.6.7 Other Local Events

None.

3.7 Enhanced Presence and MSRTC/PIDF Subscriptions Details

3.7.1 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.

MSRTC stands for the **msrtc.pidf** presence document format as explained in [MS-SIP] section 2.2.1. An **Accept** header field value of "text/xml+msrtc.pidf" is used to indicate support for this presence document format.

PIDF stands for the **pidf/rpid** presence document format as explained in [RFC3863] section 4.1 and [RFC4480] section 5. An **Accept** header field value of "application/pidf+xml" is used to indicate support for this presence document format.

The server (2) MUST be capable of handling both of these types of subscriptions. The server MUST convert enhanced presence categories to these document formats using the rules defined in this section.

A server maintains the following elements of state for each subscriber:

SubscriptionDialogEntry: Information about an individual MSRTC or PIDF subscription dialog. This information includes the URI of the subscriber. This entry also holds the subscription type information, either MSRTC or PIDF. The **SubscriptionDialogEntry** is a SIP SUBSCRIBE dialog, as defined in [RFC3265] section 3.

3.7.2 Timers

None.

3.7.3 Initialization

None.

3.7.4 Higher-Layer Triggered Events

When a PIDF/RPID Subscription request is received from a UAC, the UAS processes individual PIDF/RPID Subscription requests as specified in section 3.7.5.1.

3.7.5 Message Processing Events and Sequencing Rules

Except as specified in this section, the rules for message processing are as specified in [RFC3261] and [RFC3265].

3.7.5.1 Processing a PIDF/RPID Subscription Request

The server MUST be capable of handling PIDF/RPID SUBSCRIBE requests. For values for various headers in the SUBSCRIBE request, see section 2.2.2.8.

The PIDF/RPID SUBSCRIBE request creates a dialog with the server, and as long as the dialog is kept alive, the server sends presence updates in NOTIFY or BENOTIFY requests, as defined in [MS-SIP] section 3.5.1.1.

When an incoming dialog creating SUBSCRIBE is to be terminated as PIDF, the server creates a **SubscriptionDialogEntry** for this subscription and sets the type to "PIDF". For overview details about PIDF/RPID support, see section 1.3.3. The server then uses the logic in section 3.2.5.3 to determine the correct category publications to return for the subscription request, based on the publisher access levels. Access level is set by the publishers using their containers. While resolving which container to use, the server treats a PIDF subscription request as if it was an enhanced presence category subscription request to the **legacyInterop** category, because the server MUST generate the PIDF document, through the conversion specified in section 3.7.5.4, using the information in this category.

The server then MUST generate NOTIFY requests whenever **legacyInterop** publications in the resolved container change.

Section 4.10 includes an example that a federated user alice@fabrikam.com subscribes to bob@contoso.com's presence with PIDF subscription request.

3.7.5.2 Processing an MSRTC Subscription Request

The server MUST be capable of handling MSRTC SUBSCRIBE requests. For values for various headers in the SUBSCRIBE request, see section 2.2.2.9.

The MSRTC SUBSCRIBE request creates a dialog with the server, and as long as the dialog is kept alive, the server sends presence updates in NOTIFY or BENOTIFY requests, as defined in [MS-SIP] section 3.5.1.1.

When an incoming dialog-creating SUBSCRIBE request to be terminated as MSRTC based on the logic described in section 1.3.3, the server creates a **SubscriptionDialogEntry** for this subscription and sets the type to "MSRTC". The server then uses the logic described in section 3.2.5.3 to determine the correct category publications to return for the subscription request based on the publisher access levels. Access level is set by the publishers using their containers. While resolving which container to use, the server treats the MSRTC subscription request the same way that it treats an enhanced presence category subscription request to the **legacyInterop** category, because the server MUST generate the MSRTC document through the conversion specified in section 3.7.5.5, using the information in this **category** instance.

The server then MUST generate NOTIFY requests whenever **legacyInterop** publications in the resolved container change.

Section 4.11 includes an example that a federated user alice@fabrikam.com subscribes to bob@contoso.com's presence with MSRTC subscription request.

3.7.5.3 Maintain MSRTC or PIDF Subscriber List Entry

The subscriber list is described in section 2.2.2.6, and specified in section 3.6.

When the incoming SUBSCRIBE request is MSRTC or PIDF, as opposed to category SUBSCRIBE as explained in preceding sections, the subscriber list-related call flow does not change. These subscriptions cause subscribers to be inserted into the publisher's subscriber list if they are not already present there. The main difference is that MSRTC and PIDF subscriptions do not have the "context" concept, which gives category subscribers control over getting themselves into the publisher's subscriber list. Because of this difference, removal of a **SubscriberEntry** when the **acknowledged** attribute is "true" is not possible in the case of MSRTC and PIDF subscriptions. The server MUST NOT remove the subscriber entries for these subscriptions. Therefore, even when MSRTC and PIDF subscribers are acknowledged through a **setSubscriber** request, they are always kept in the subscriber list with the **acknowledged** attribute value set to "true".

3.7.5.4 Generating a NOTIFY Response to a PIDF SUBSCRIBE Request

The server MUST generate a NOTIFY request in response to a PIDF subscription request. In addition, each operation that causes a change in the **legacyInterop category** instance or instances in the container that the request resolved to, generates a NOTIFY response on a PIDF subscription dialog. An example of such an operation is an endpoint that publishes a new presence state that changes the value of a **legacyInterop category** instance by the computing function. These operations can also be server actions, such as the expiration of publications.

During the PIDF NOTIFY generation process, the server checks all **SubscriptionDialogEntry** instances of type PIDF that involve this publisher and sends a NOTIFY response to the subscribing endpoints of the subscriber with the new data.

The server retrieves the **legacyInterop category** instance from a resolved container for this subscription. For details, see section 2.2.2.7.6.

The server MUST create the XML body for the PIDF NOTIFY response. For details about the format of this XML, see section 2.2.2.8.

The server MUST use the **availability** and **token** attribute values from the **legacyInterop category** instance (section 2.2.2.7.6) to create **PIDF Availability** and **RPID Activity** elements as shown in the following table.

Enhanced presence availability	Enhanced presence activity	PIDF availability	RPID activity
0-to-2999	Any	Closed	[none]
3000-to-4499	Any	Open	[none]
4500-to-5999	Any	Open	Away
6000-to-7499	Any-but-On the Phone	Open	Busy
6000-to-7499	On the Phone	Open	On the phone
7500-to-8999	Any	Open	Away
9000-to-11999	Any	Open	Busy
12000-to-14999	Any	Open	Away
15000-to-17999	Any	Open	Away
>= 18000	Any	Closed	[none]

The server SHOULD set the *ms-remote-fqdn* parameter in the ms-edge-proxy-message-trust header in the NOTIFY request. The client SHOULD look up the *ms-remote-fqdn* value in the publicProviders provision group and find the name and iconUrl for the network which the remote user belongs to.

The server sends a NOTIFY request with the presence of the user, for example bob@contoso.com. An example of a PIDF/RPID NOTIFY is found in section 4.10.

The subscribing SIP protocol client acknowledges the receipt of the notification with a 200 OK.

3.7.5.5 Generating a 200 OK or NOTIFY Response to an MSRTC SUBSCRIBE Request

The server MUST send out a 200 OK with the presence document if the **Supported** header field in the subscription request contains "ms-piggyback-first-notify". For details about "ms-piggyback-first-

notify", see [MS-SIP] section 3.4.5.1. Otherwise, the server MUST generate a NOTIFY request in response to an MSRTC subscription request.

In addition, each operation that causes a change in the **legacyInterop category** instance or instances in the container that the request resolved to, generates a NOTIFY response on an MSRTC subscription dialog. An example of such an operation is an endpoint that publishes a new presence state that changes the value of a **legacyInterop category** instance by the computing function. These operations can also be server actions, such as the expiration of publications.

While the MSRTC NOTIFY response is generated, the server checks all **SubscriptionDialogEntry** instances of type **MSRTC** that involve this publisher and sends a NOTIFY response to the subscribing endpoints of the subscriber with the new data.

A sample 200 OK response to a SUBSCRIBE request is found in section 4.11.

The Content-Type header field MUST be set to "text/xml+msrtc.pidf".

The server retrieves the **legacyInterop category** instance from a resolved container for this subscription. For details, see section 2.2.2.7.6.

The server MUST create the XML body for the MSRTC NOTIFY response with the following elements and attributes. For details on the format of this XML, see section 2.2.2.9.

availability: The server SHOULD generate this element with the following attribute.

> **aggregate:** The server SHOULD generate the value of this attribute as shown in the table that follows this list.

activity: The server SHOULD generate this element with the following attribute.

> **aggregate:** The server SHOULD generate the value of this attribute as shown in the table that follows this list.

displayName: The server SHOULD generate this element with the following attribute.

> **displayName:** Display name of the publisher. The server MAY get the value of this attribute from a Directory Service (DS)<39>.

email: The server SHOULD generate this element with the following attribute.

> **email:** E-mail address of publisher. The server MAY get the value of this attribute from a DS<40>.

phoneNumber: The server SHOULD generate this element with the following attribute.

> **number:** Phone number of publisher. The server MAY get the value of this attribute from a DS<41>.

aggregate: The server SHOULD generate this element with the **states** subelement.

state: The server SHOULD generate this element with the following attributes. The server MUST generate the value of this element as shown in the table that follows this list. The return value MUST equal the token value retrieved from the **legacyInterop category** instance.

> **avail:** The server SHOULD generate the value of this attribute as shown in the table that follows this list.
>
> **xsi:type:** The value SHOULD be "userState".

The server MUST use the **availability** and **token** attribute values from the **legacyInterop category** instance to create MSRTC elements and attributes as shown in the following table.

Availability value retrieved from legacyInterop category instance	Token value retrieved from legacyInterop category instance	Value of avail attribute	Value of state element	Value of aggregate attribute in availability element	Value of aggregate attribute in activity element
0 to 2999	Any	18500	Any	0	100
3000 to 4499	Any	3500	Any	300	400
4500 to 5999	Any	15500	Any	300	100
6000 to 7499	Any but on-the-phone	6500	Any but on-the-phone	300	600
6000 to 7499	on-the-phone	6500	on-the-phone	300	500
7500 to 8999	Any	15500	Any	300	100
9000 to 11999	Any	9500	Any	300	600
12000 to 14999	Any	12500	Any	300	300
15000 to 17999	Any	15500	Any	300	100
Equal to or greater than 18000	Any	18500	Any	0	100

3.7.5.6 Error Responses

None.

3.7.6 Timer Events

None.

3.7.7 Other Local Events

None.

3.8 Enhanced Presence Aggregation Details

The process of aggregation is modeled as a transformation function. The function computes a single value of a category based only on instances of one or more different categories that have previously been published. After this category's value has been computed, it can be published to multiple containers, and it can be subscribed to, in the same way as any other category.

The server can determine which published categories the computed category is dependent on. When any of these dependent categories are published, the server calls the transformation function to recompute the new aggregate category value.

Each container can have a computed aggregate value that is published by the transformation function. The computed value can be different for different containers, but the transformation function that produces a given category value is identical for any specified container. The transformation function is run for each specified container, but these calls can differ only by the input values that each container supplies to the function.

Each container has attributes that indicate whether it computes any categories and if so, what those categories are. That is, a container can specify zero, one, or more computed categories.

The specification of which categories are computed for which containers is not on a per-publisher basis: it is global. If a category called "state" with a particular instance is computed for containers 2 and 3, it is computed for all publishers who have those containers. Protocol clients do not normally publish computed categories to such containers; however, a category that is flagged as being computed in one container can be published in a different container as long as it is not flagged as being computed there either.

One example of this is the **state** category. **State** is not computed in the default container so that a persistent default can be published by a SIP protocol client to this container. This value is then given to subscribers if the user is offline and has had no publications to other containers with which to compute the **state** category value in those containers.

Generally, the input **category** instances given to the transformation function come from the same container for which the aggregate category is to be computed. All instances of these input categories from the container for which the value is being computed are provided to the transformation function. Thus, the function needs to be prepared to receive not just one value for each input category, but an array of values for each **category**.

For example, calendar data can be published only to container 40, because members of that container are allowed to see calendar data, whereas members of containers 10 and 20 are not. However, this calendar data implicitly affects the aggregate state values computed for containers 10 and 20. The transformation function that computes these values in containers 10 and 20 needs access to the calendar data from container 40, because the data is not published in the lower-numbered containers.

To summarize, the transformation function receives a table as input. Each row of this table identifies one of the following unique publications:

- Container number
- **Category** name
- Instance number
- Last publish time
- **Category** data value

For the categories meant for self-consumption subscriptions to them from other users are not allowed. The main purpose for this restriction is to allow endpoints to be able to use enhanced presence as a mechanism to roam data for their own use.

For the categories marked as read-only in containers, the server cannot allow SIP protocol clients to overwrite or delete these categories in those containers. An example of a read-only **category** is the **userProperties category** in container 1. This restriction is used for protecting server publications. For details about server publications, see section 3.2.6.2.

3.8.1 Abstract Data Model

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

The abstract data model used in enhanced presence aggregation is defined in section 3.2.1.

3.8.2 Timers

None.

3.8.3 Initialization

None.

3.8.4 Higher-Layer Triggered Events

None.

3.8.5 Message Processing Events and Sequencing Rules

3.8.5.1 Aggregation of State Category

An enhanced presence server aggregates the **state** category as specified in section 2.2.2.7.1 to produce the user's overall presence state, or **aggregateState** and **aggregateMachineState** instance types, as specified in the figure titled Aggregation of user's presence state in section 3.8.5.1.1.

3.8.5.1.1 Overview

The following figure specifies the overview of aggregating a user's overall presence state. Elements in the figure are explained in detail following the figure.

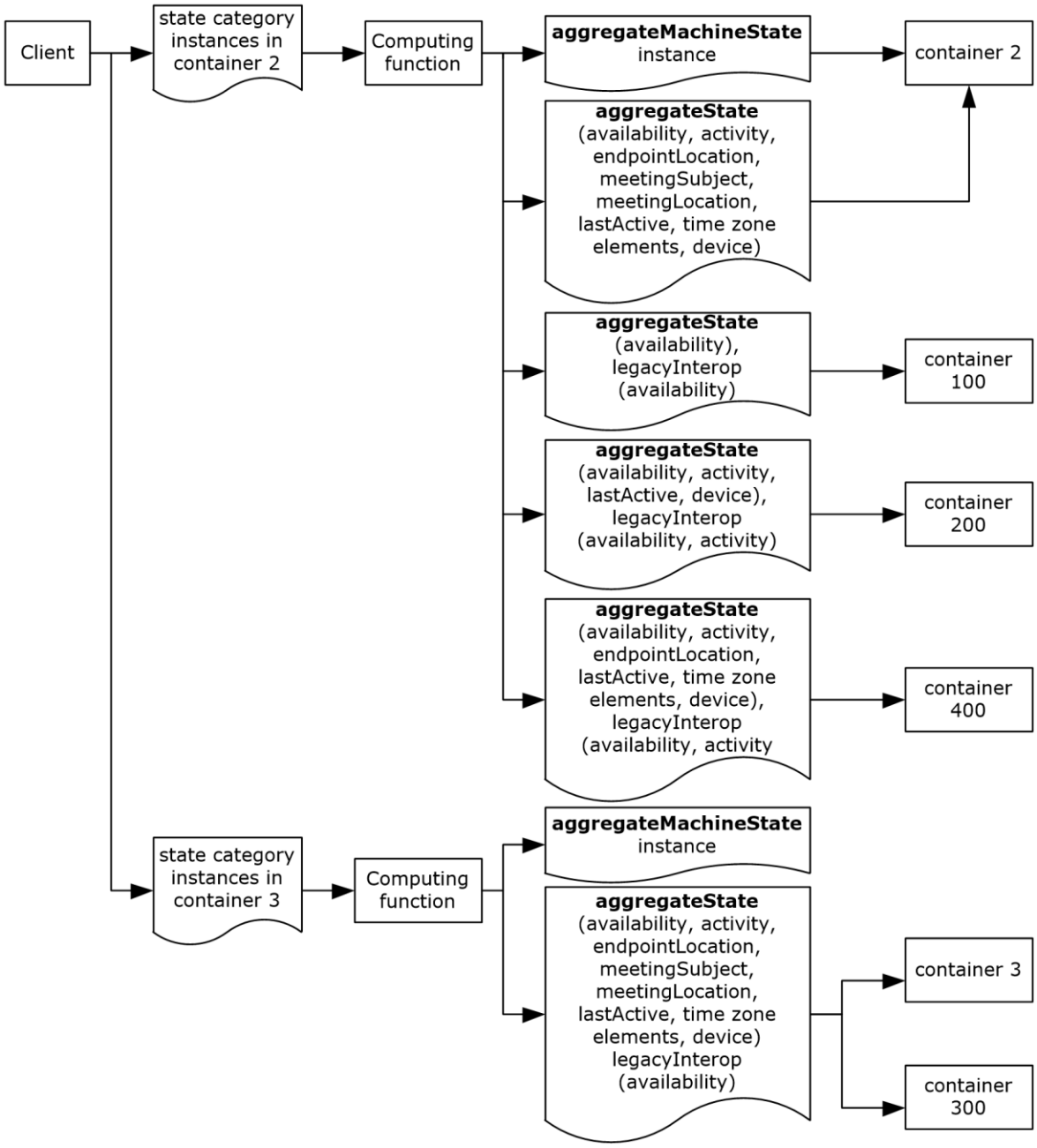

Figure 3: Aggregation of user's presence state

SIP protocol client: The SIP protocol client publishes **state category** instances into containers 2 and 3. Although the server does not prevent it, the SIP protocol client MUST NOT add any members to containers 2 and 3.

Computing function: The computing function in the server MUST be triggered if there are any publications to the **state category** in containers 2 or 3. The computing function MUST compute and publish state, both **aggregateState** and **aggregateMachineState**, and **legacyInterop category** instances into other containers, as shown in section 3.8.5.1.2.7.

legacyInterop category: The **legacyInterop category** MUST be registered as a private category. The server MUST use this **category** instance in the containers to generate NOTIFY requests for PIDF and MSRTC subscriptions. For details, see section 3.7.5.4, and section 3.7.5.5.

Publications to containers: Different flavors of **aggregateState** as specified in section 2.2.2.7.1 and **legacyInterop** instances MUST be published into different containers, as shown in section 3.8.5.1.2.7. The SIP protocol client MAY place members in those containers to control the amount of information given to subscribers of those categories. The computing function can optimize not to publish anything if an existing instance is identical to a new instance in a given container.

3.8.5.1.2 High Level Call Flow

The following figure illustrates the high level call flow of aggregating a user's overall presence state. Each element in the figure is explained in sub sections following the diagram.<42>

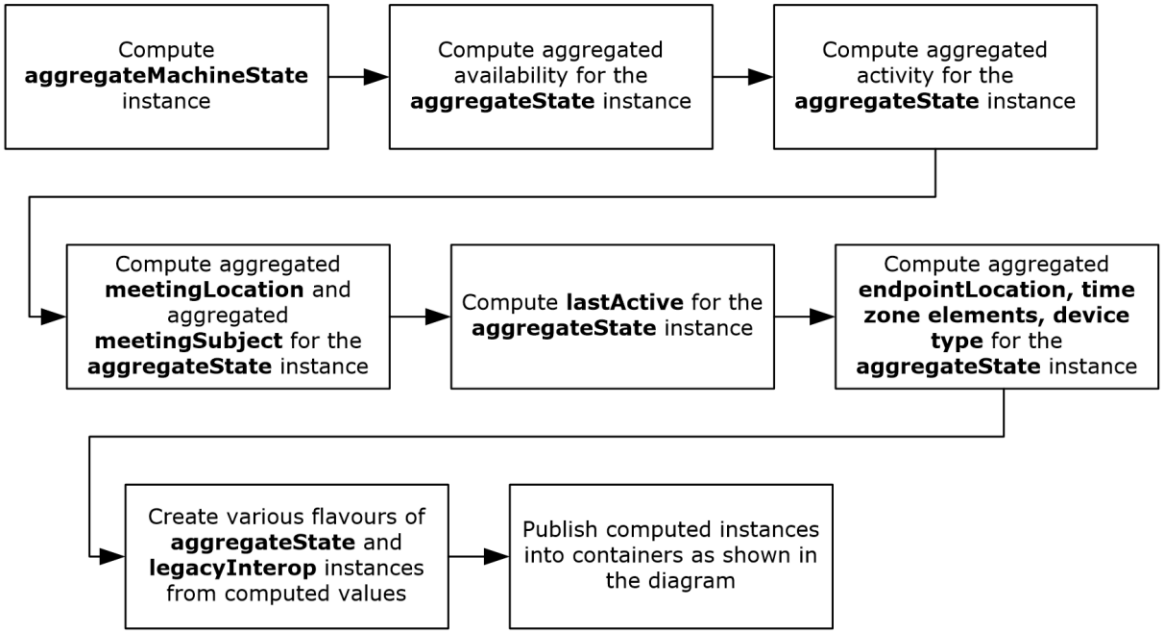

Figure 4: High-level call flow

3.8.5.1.2.1 Computation of aggregateMachineState instance

The following figure specifies the process of computing **aggregateMachineState** instance, as specified in section 2.2.2.7.1.

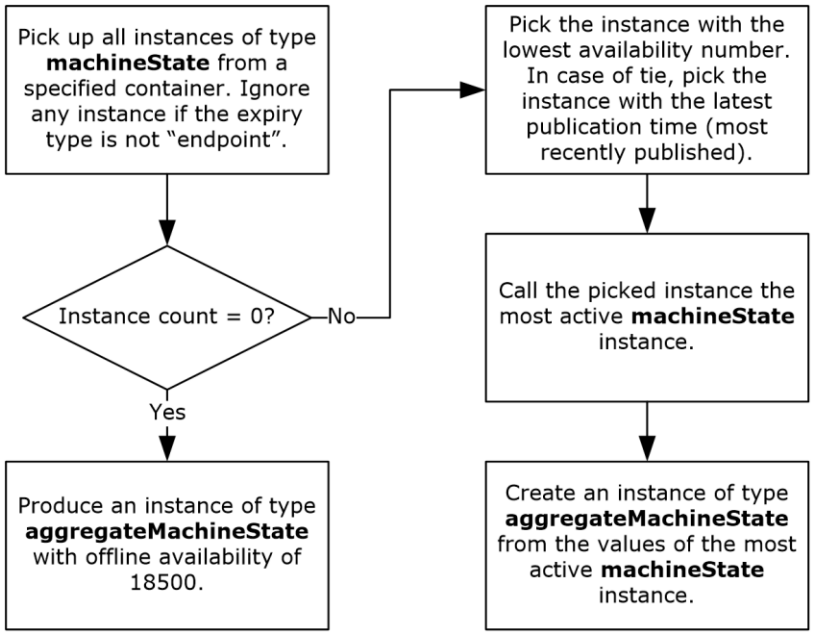

Figure 5: Computation of aggregateMachineState instance

The computing function picks up all instances of type **machineState** from a specified container. It ignores any instance if the expiry type is not "endpoint". If there are no instances, the computing function MUST produce an instance of type **aggregateMachineState** with an offline availability of 18500.

If there are any instances of type **machineState**, the computing function picks the **machineState** instance with the lowest availability number as the most active **machineState** instance. If there are multiple instances that have the same availability, the **machineState** instance that published most recently is used. For example, if one **machineState** instance has an availability of 3500 that was published at 1:00 PM, and another has an availability of 3500 that was published at 1:03 PM, the second **machineState** instance is used. An **aggregateMachineState** instance is created based on the values from the most active **machineState** instance.

3.8.5.1.2.2 Computation of Aggregated Availability

The following figure specifies the process of computing aggregated availability.

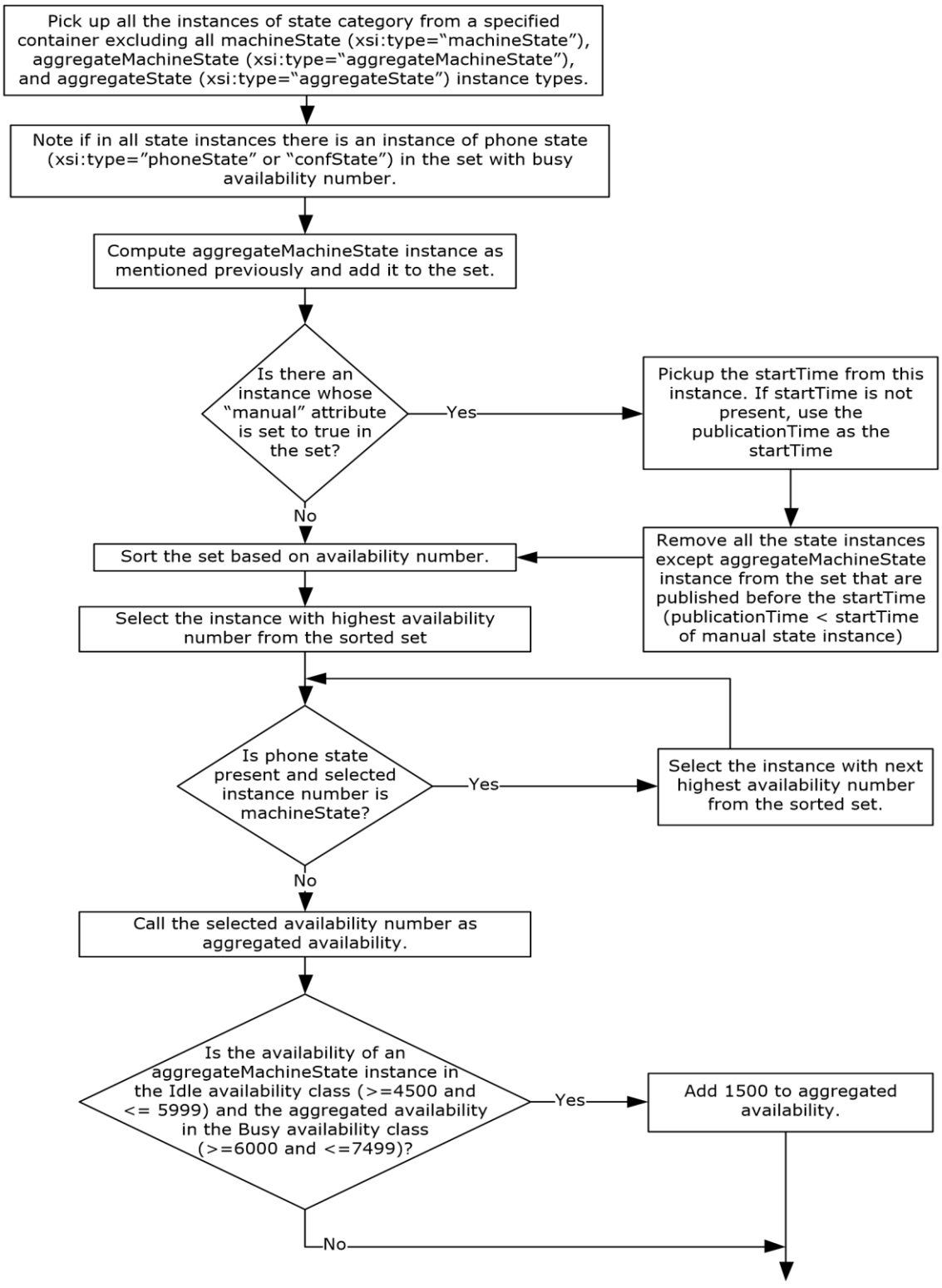

Figure 6: Computation of aggregated availability

To calculate aggregated availability, the computing function gets the set of all state instances from a given container. **machineState**, **aggregateMachineState**, and **aggregateState** instance types are excluded from the set. An **aggregateMachineState** instance is computed as mentioned previously and added to the set.

If there is a state instance whose **manual** attribute is set to "true", the computing function removes all state instances older than this state instance. The computing function uses the **startTime** attribute to determine the age of the state instance and uses the **publicationTime** of the state **category** instance if **startTime** is not present. The **aggregateMachineState** instance is excluded in this filtering step.

The computing function extracts the greater availability number from the set as the aggregated availability.

If the availability of an **aggregateMachineState** instance is in the Idle availability class, which is between 4500 and 5999, and the aggregated availability is in the Busy availability class, which is between 6000 and 7499, the computing function adds 1500 to the aggregated availability. This is the availability of the **aggregateState** instance.

3.8.5.1.2.3 Computation of aggregated activity

The following figure specifies the process of computing aggregated activity.<43>

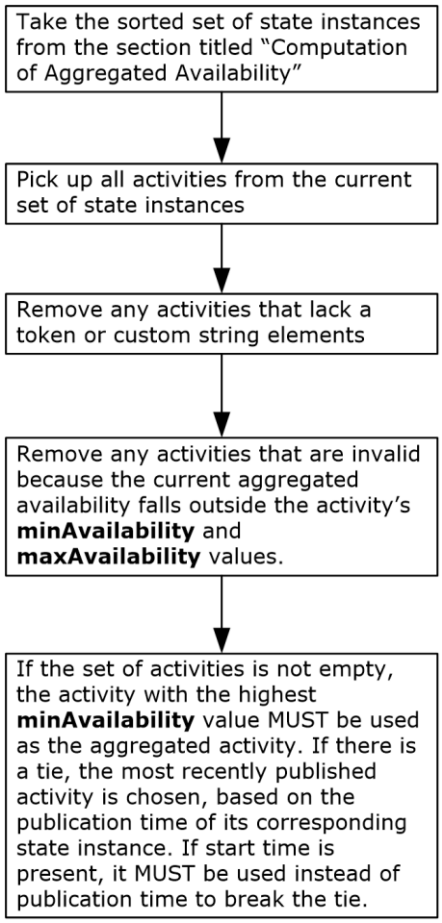

Figure 7: Computation of aggregated activity

To calculate aggregated activity, the computing function uses the sorted set of **state** instances from the end of the previous aggregated availability step.

The computing function begins with all of the activities from the current set of state instances. It removes any activities that lack a token or custom string, as well as any activities that are invalid because the current aggregate availability does not fall between the activity's **minAvailability** and **maxAvailability** values.

If an activity set is not empty, the activity with the highest **minAvailability** value MUST be used as the current activity. In case of a tie, the most recently published activity, as determined by the publication time of the corresponding **state** instance, is used. If start time is present, it MUST be used instead of publication time to break the tie. Aggregated activity SHOULD<44> be empty if the set of activities is empty.

3.8.5.1.2.4 Computation of lastActive for the aggregateState instance

The following figure specifies the process of computing lastActive for the **aggregateState** instance.

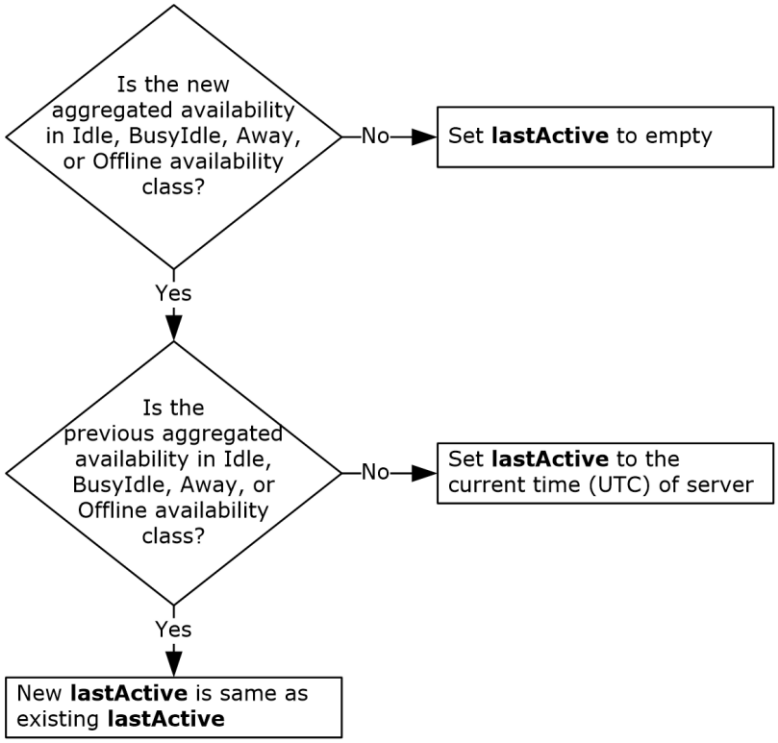

Figure 8: Computation of lastActive for the aggregateState instance

3.8.5.1.2.5 Computation of Aggregated meetingLocation and Aggregated meetingSubject for the aggregateState Instance

If a **calendarState** instance type exists with either **meetingLocation** or **meetingSubject** elements, the computing function MUST use those elements as aggregated **meetingLocation** and aggregated **meetingSubject**, respectively. If there are multiple **calendarState** instances with these elements, the aggregated **meetingSubject** and aggregated **meetingLocation** elements MUST be set to empty.

3.8.5.1.2.6 Computation of Aggregated endpointLocation, time zone elements, and device type for the Aggregated state Instance

If the aggregated availability is less than 12000, the computing function SHOULD<45> pick up the **endpointLocation** and **time zone** elements, and the **device type** from the **aggregateMachineState** and use them as the aggregated **endpointLocation, time zone** elements**,** and **device type** for the **aggregateState** instance.

If the aggregate availability is greater than or equal to 12000, the computing function can ignore the **endpointLocation**, **time zone** elements, and **device type** from the **aggregateMachineState** when calculating the **aggregateState** instance.

3.8.5.1.2.7 Publication of computed category instances into containers

The following table describes the relationship of containers used as computational inputs with containers that hold the results of the computation. For example, a **state category** instance published to the Input Container 3 is computed to be placed in the Output Container 300.

In addition, the table defines the set of state attributes (Instance Data) of the placed state category instance, as specified in section 2.2.2.7.1, for each output container. Finally, the expiration type (expireType) and instance Id of the **state category** instance are specified for each output container.

Input container	Output container	Category name	expireType, and instance attribute values	Instance data
2	100	state	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	xsi:type – aggregateState availability – "Aggregated availability"
2	100	legacyInterop	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	availability – "Aggregated availability"
2	200	state	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	xsi:type – aggregateState availability – "Aggregated availability" token – "Aggregated activity" lastActive, device - device type
2	200	legacyInterop	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	availability – "Aggregated availability"
3	300	state	user, 1 - If there are any published states of	xsi:type – aggregateState availability – "Aggregated availability"

Input container	Output container	Category name	expireType, and instance attribute values	Instance data
			type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	token – "Aggregated activity" endpointLocation – "Aggregated endpointLocation" meetingSubject – "Aggregated meetingSubject" meetingLocation – "Aggregated meetingLocation" lastActive timeZoneBias - "Aggregated timeZoneBias" timeZoneName - "Aggregated timeZone Name" timeZoneAbbreviation = "Aggregated timeZoneAbbreviation " device - device type
3	300	legacyInterop	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	Availability - "Aggregated availability"
2	400	state	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	xsi:type – aggregateState availability – "Aggregated availability" token – "Aggregated activity" endpointLocation – "Aggregated endpointLocation" lastActive timeZoneBias - "Aggregated timeZoneBias" timeZoneName - "Aggregated timeZone Name" timeZoneAbbreviation = "Aggregated timeZoneAbbreviation " device - device type
2	400	legacyInterop	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	availability – "Aggregated availability"
2	2	state	user, 0x10000000	xsi:type – aggregateMachineState availability, activity, endpointId– These element/attributes are copied from the most active machineState instance.

Input container	Output container	Category name	expireType, and instance attribute values	Instance data
2	2	state	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	xsi:type – aggregateState availability – "Aggregated availability" token – "Aggregated activity" endpointLocation – "Aggregated endpointLocation" meetingSubject – "Aggregated meetingSubject" meetingLocation – "Aggregated meetingLocation" lastActive timeZoneBias - "Aggregated timeZoneBias" timeZoneName - "Aggregated timeZone Name" timeZoneAbbreviation = "Aggregated timeZoneAbbreviation " device - device type
3	3	state	user, 1 - If there are any published states of type xsi:type:machineState. static, 0 - If there are no published states of type xsi:type:machineState.	xsi:type – aggregateState availability – "Aggregated availability" token – "Aggregated activity" endpointLocation – "Aggregated endpointLocation" meetingSubject – "Aggregated meetingSubject" meetingLocation – "Aggregated meetingLocation" lastActive timeZoneBias - "Aggregated timeZoneBias" timeZoneName - "Aggregated timeZone Name" timeZoneAbbreviation = "Aggregated timeZoneAbbreviation " device - device type

3.8.5.2 Aggregation of Device Category

An enhanced presence server aggregates the **device category** to produce a **services category** that carries the aggregated set of capabilities for a publisher, as specified in the following sections.

3.8.5.2.1 Overview

The following figure specifies the overview in aggregating a **device category**. Elements of the figure are explained in the rest of this section.

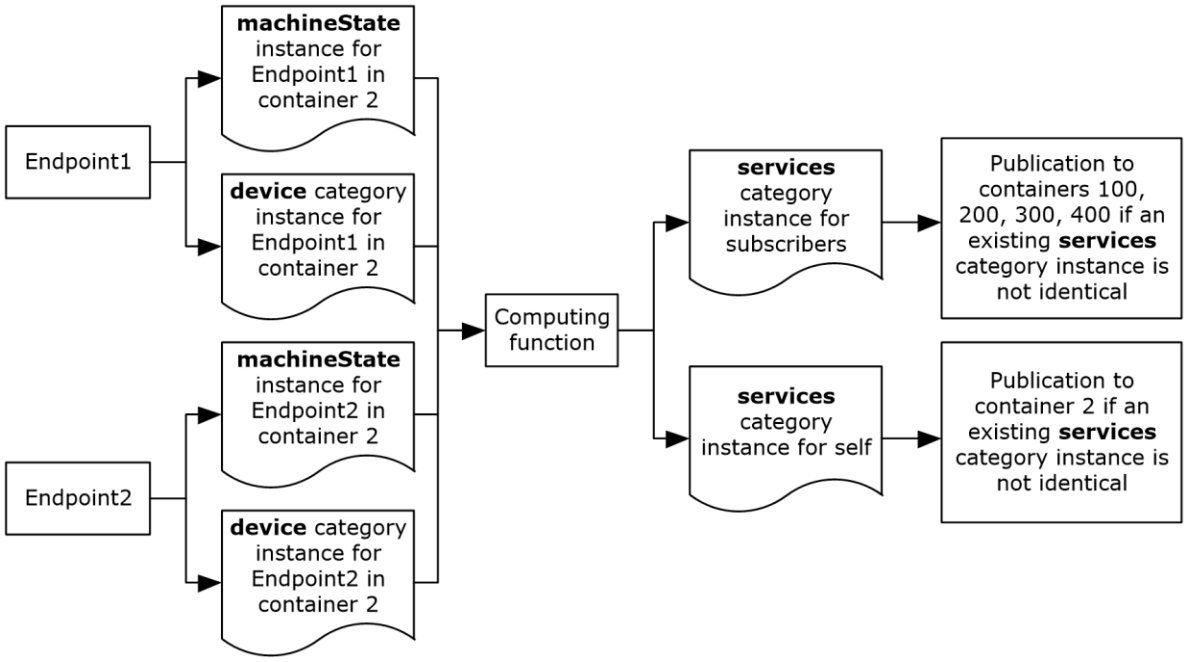

Figure 9: Aggregation of a device category

SIP protocol client: The SIP protocol client publishes the **machineState** instance of its endpoint and the capabilities of the endpoint in the **device category** instance to container 2. The SIP protocol client MUST NOT add any members to container 2. Members added to this container could subscribe to categories in the same container. Because the data in this container is specific to the publisher, this kind of subscription could compromise the privacy of the publisher.

Computing Function: The computing function in the server MUST be triggered if there are any publications to device or state categories in container 2. The computing function MUST aggregate all of the **machineState** and **device category** instances in container 2 to produce **services category** instances.

Publication to Containers: The computing function MUST publish a **services category** instance for self into container 2 and a **services category** instance for subscribers into containers 100, 200, 300, and 400. The computing function can optimize by not publishing anything if an existing instance is identical to a new instance in a given container. The computing function SHOULD get existing instances of the **services category** from container 2 to implement such an optimization.

3.8.5.2.2 High-Level Call Flow

The following figure specifies the logic flow involved in aggregating a **device category**.

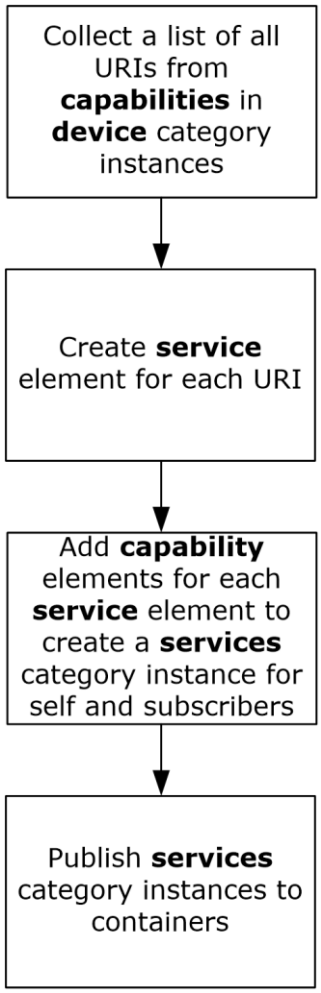

Figure 10: High-level call flow

The first step in **services** aggregation is collecting a list of all URIs from capabilities in **device category** instances. A separate **service** element is created for each URI. Capabilities are filled in for each **service** element. A **services category** instance for self and subscribers are created from these **service** elements.

URI Set: The computing function iterates through each **device category** instance from container 2 and records the **uri** attribute of each **capability** element. This produces the set of all URIs. All duplicates are eliminated from the URI set.

service Element for Each URI: Each of the URIs in the preceding set MUST correspond to a **service** element. The computing function MUST generate one **service** element per URI.

Create services Category Instances for Self and Subscribers: The computing function SHOULD use the following logic to create **services category** instances for self and subscribers. This logic walks through all the URIs and performs the specified steps for each URI in the URI set.

1. Get the list of capabilities from all of the **device category** instances. For more information about device capabilities within the **service category**, see section 2.2.2.7.3.

2. Sort the list of capabilities by capability type.

3. Walk through all of the capability types and perform the following steps for each capability type:

1. For each capability type, walk through all of the capability instances and calculate **winning-capability** and **winning-machineState**. For example, one endpoint might publish a **text** capability instance with its **render** and **capture** attributes set to "true", while another endpoint might publish a **text** capability with the same attributes set to "false".

1. Get the **endpointId** from the **device** instance of the current capability.

2. Get the **machineState** instance that corresponds to the **endpointId** of the previous step.

3. If any of the following cases is true, choose the current capability as the **winning-capability** and choose the current **machineState** instance as the **winning-machineState**:

- **winning-capability** is empty.
- The capability contains a **publish** attribute set to "true", and the following case is true:
- The **version** attribute of the current capability is greater than that of **winning-capability**.
- The capability contains a **publish** attribute set to "false", and any of the following cases are true:
- The **render** and **capture** attributes of **winning-capability** are set to "false".
- The **render** and **capture** attributes of **winning-capability** are not set to "false", and the current **machineState** is more active than that of **winning-machineState**.

> **Note**: The current machine state is considered to be more active than the **winning-machineState** if the availability of the current machine state is less than that of **winning-machineState** or if the availability of the current machine state is equal to that of **winning-machineState** and the publication time of the current **machineState** instance is greater than that of **winning-machineState**.

2. Add the availability from the **winning-machineState** as the **deviceAvailability** attribute to the **winning-capability**.

3. If the **publish** attribute is set to "false", add the **winning-capability** to the **service** element of **URI** in the **services** instance for subscribers.

4. Add **winning-capability** to the **service** element of **URI** in the **services** instance for self. Also, add the **endpointId** of **winning-machineState** as the **preferredEndpointId** attribute to the winning capability.

Publication of services Category Instances into Containers: The computing function MUST generate the outputs as shown in the following table.

Input container	Output container	Category name	expireType	instance	Instance data
2	100	services	user	0	**service** elements with all capabilities that contain **render** or **capture** attributes. Each capability SHOULD have a **deviceAvailability** attribute.
2	200	services	user	0	**service** elements with all capabilities that contain **render** or **capture** attributes. Each capability SHOULD have a **deviceAvailability** attribute.

Input container	Output container	Category name	expireType	instance	Instance data
2	300	services	user	0	**service** elements with all capabilities that contain **render** or **capture** attributes. Each capability SHOULD have a **deviceAvailability** attribute.
2	400	services	user	0	**service** elements with all capabilities that contain **render** or **capture** attributes. Each capability SHOULD have a **deviceAvailability** attribute.
2	2	services	user	0	**service** elements with all the capabilities that contain **render** or **capture** or **publish** attributes. Each capability SHOULD have **preferredEndpointId** and **deviceAvailability** attributes.

3.8.5.3 Creation of workingHours category

This section describes behaviors supported in versions described by endnote <46>.

An enhanced presence server creates a **workingHours category** instance by extracting working hours information from the instance zero of the **calendarData category** in container 1, as specified in the following diagram.

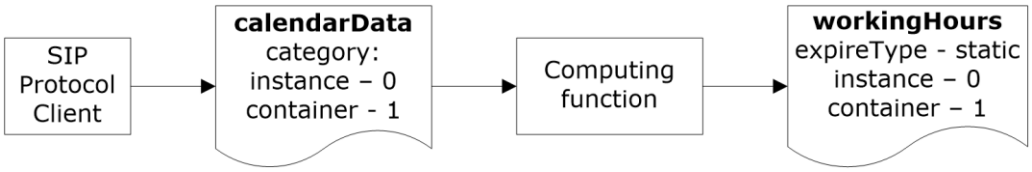

Figure 11: Creating a workingHours category instance

SIP protocol client: The SIP protocol client MUST publish instance zero of the **calendarData category** into container 1. Although the server does not prevent it, the SIP protocol client MUST NOT add any members to container 1. When publishing the **calendarData category** instance, the SIP protocol client MUST compare existing data with the new data to avoid publication if there are no changes.

Computing function: The computing function in the server MUST be triggered if there are any publications to the **calendarData category** in container 1. The computing function MUST extract the **WorkingHours** element from instance zero of the **calendarData category** in container 1 to create and publish instance zero of the **workingHours category** in container 1. If **calendarData category** is unpublished from container 1 or **workingHours** element is removed from **calendarData category**, computing function MUST unpublish the **workingHours** category from container1. The computing function MUST compare existing data with new data to avoid publication if there are no changes. For examples of the **workingHours category** instance generation, see section 4.3.3.

3.8.5.4 Creation of dndState category

This section describes behaviors supported in versions described by endnote <47>.

An enhanced presence server creates a **dndState category** instance based on a user state, **xsi:type** equals "userState", instance of the **state category**, or a presenting state, **xsi:type** equals "presentingState", instance of the **state category**, as shown in the following figure.

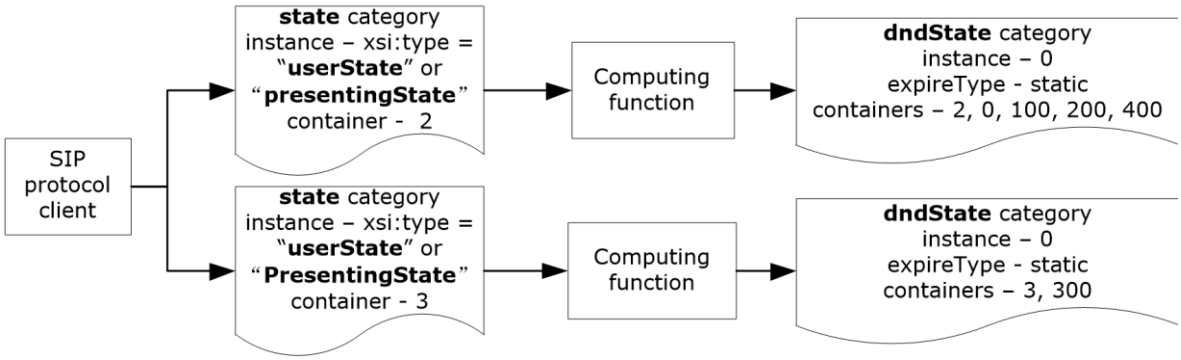

Figure 12: Creating a dndState category instance

SIP protocol client: The SIP protocol client MUST publish **state category** instances into container 2 and 3. Although the server does not prevent it, the SIP protocol client MUST NOT add any members to container 2 and 3.

Computing function: The computing function in the server MUST be triggered if there are any publications to the **state category** in containers 2 and 3. The computing function MUST look at the user state (**xsi:type** equals "userState") and presenting state (**xsi:type** equals "presentingState") instances of the **state category** in container 2 to create and publish instance zero of the **dndState category** in containers 2, 0, 100, 200 and 400. Similarly, the computing function MUST look at the user state (**xsi:type** equals "userState") and presenting state (**xsi:type** equals "presentingState") instances of the **state category** in container 3 to create and publish instance zero of the **dndState category** in containers 3 and 300. The computing function MUST compare existing data with the new data to avoid publication if there are no changes. The computing function MUST treat the **state** instance as user state if the **xsi:type** attribute value of the **state** element in the **state category** instance matches **userState**. The computing function MUST treat the **state** instance as presenting state if the **xsi:type** attribute value of the **state** element in the **state category** instance matches **presentingState**. For more details on the format of the **state category**, see section 2.2.2.7.1. The computing function MUST create a **dndState category** instance with **availability** "9500" if the **availability** of the **userState** or **presentingState** instance of the **state category** falls in the "**Do Not Disturb**" range (9000 to 11999). Similarly, the computing function MUST create a **dndState category** instance without the **availability** element if the **availability** of a **userState** and **presentingState** instance of the **state category** falls outside of the **Do Not Disturb** range (9000 to 11999) or **userState** and **presentingState** instance of **state category** are not present. For examples of the **dndState category** instance generation, see section 4.3.4.

3.8.5.5 Error Responses

None.

3.8.6 Timer Events

None.

3.8.7 Other Local Events

None.

3.9 Delegate Management Details

For details about delegate management, see section 1.3.4.

3.9.1 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.

An enhanced presence server maintains the following elements of state for each **delegator**:

DelegateSet: A set of delegate entries. Entries are keyed on the URI of the delegate. The set has an associated version number that is used to synchronize updates from multiple endpoints of the same delegator.

DelegateEntry: Information about an individual delegate. Each entry contains Boolean values that store the value provided for the **publishOperation** and **redelegateOperation** elements in the **setDelegate** XML schema defined in section 6.14.

3.9.2 Timers

None.

3.9.3 Initialization

None.

3.9.4 Higher-Layer Triggered Events

When a **setDelegate** request is received from a UAC, the UAS processes individual **setDelegate** requests as specified in section 3.9.5.1.

3.9.5 Message Processing Events and Sequencing Rules

Except as specified in this section, the rules for message processing are as specified in [RFC3261] and [RFC3265].

3.9.5.1 Processing a setDelegate Request

Enhanced presence uses a **setDelegate** SERVICE request for modifications to the delegate list. Delegators can operate, including add, modify or remove, on one delegate at a time with the server.

3.9.5.2 Processing Requests

In the example found in section 4.12, bob@contoso.com is adding alice@contoso.com to the delegate list.

As specified in section 2.2.2.10.2, a **setDelegate** request is identified by a SERVICE request with a Content-Type header field value of "application/msrtc-setDelegate +xml". Upon receipt of such a message, the server MUST verify that the **To-URI** and the **From-URI** in the SERVICE header fields are identical. The **To-URI** is then used to look up the delegator whose delegate list is to be modified.

The server MUST validate the different attributes in the request to conform to the XML schema specified in section 2.2.2.10.2. The server then looks up the **DelegateSet** for this **Delegator**. The first step in processing the request is to validate the version. The specified version MUST equal the version number associated with the **DelegateSet**. The DelegateSet is empty when there are no delegates for the delegator and the version number associated with the DelegateSet is zero if this is the first action on the DelegateSet.

The server then starts processing the **delegate** element. The server validates the specified **uri** and **action**. The **action** MUST be "add", "modify" or "remove". The **uri** MUST NOT be identical to the delegator URI. The server MUST validate that the **uri** is assigned to a valid presentity in this deployment. If child elements of the delegate are present and they are either **publishOperation** or **redelegateOperation**, the server validates the authorization specified in these child elements. The authorization MUST be either "allow" or "deny". Absence of the element implies "deny" for the authorization. The server then looks up the **DelegateEntry** based on the **uri**.

If the **delegate** element has an **action** of "add" or "modify" the server creates a new **DelegateEntry** or updates the existing **DelegateEntry**, sets the Boolean values to the authorization specified in the child elements, and inserts it into the **DelegateSet**. If the **action** is "remove", the server removes the existing **DelegateEntry** from the **DelegateSet** and then deletes it. A request to "add" a **DelegateEntry** that exists or "modify" a **DelegateEntry** that does not exist is valid and MUST be accepted. When a request to "add" a DelegateEntry that exists comes in, the server updates the existing **DelegateEntry** and sets the Boolean values to the authorization specified in the child elements of the new request. When a request to "modify" a DelegateEntry that does not exist comes in, the server creates a new **DelegateEntry** and sets the Boolean values to the authorization specified in the child elements, and inserts it into the **DelegateSet.** A request to "remove" a **DelegateEntry** that does not exist is invalid and MUST be rejected.

The server SHOULD limit the total number of **DelegateEntry** elements that can exist for a given delegator. The implementation SHOULD permit an administrator to configure these limits based on the needs of the users in a deployment by some means outside the scope of this specification. The server SHOULD fail any **setDelegate** request that causes this limit to be exceeded.

When the **setDelegate** request is successfully processed, the server updates the version number associated with the **DelegateSet**. The server MUST NOT commit any changes until the request is completely processed. If the operation is successful, the server sends a 200 OK response as specified in [RFC3261] section 21.2.

3.9.5.3 Generating NOTIFY for SelfSubscribers

Continuing from the preceding example, the BENOTIFY message in section 4.12.1 illustrates a BENOTIFY generated by the server on an active self subscription dialog. [MS-SIP] section 3.5.1.1 defines the conditions under which a server can send a BENOTIFY request.

After a delegator's **setDelegate** request has been successfully processed, the server MUST generate a NOTIFY or BENOTIFY request for each corresponding **SelfSubscriptionDialogEntry**. For a **SelfSubscriptionDialogEntry** to correspond, its subscribed URI MUST be identical to the delegator's URI and its scope MUST include "delegates". The NOTIFY or BENOTIFY request MUST be generated even for the endpoint that sent the **setDelegate** request, as long as that endpoint has a corresponding **SelfSubscriptionDialogEntry**.

As illustrated in section 2.2.2.3.3, the generated NOTIFY or BENOTIFY request MUST have a Content-Type header field value of "application/vnd-microsoft-roaming-self+xml" and MUST embed a **delegates** document. The **member** elements wrapped by the **delegates** element MUST together represent the complete new membership of the **DelegateSet**. If the **DelegateSet** is now empty, the

delegates element MUST have an empty body. The **delegates** document MUST also specify the updated version number of the **DelegateSet**.

3.9.5.4 Error Responses

The server MUST send back a 400 Bad Request response if the XML body is not a valid **http://schemas.microsoft.com/2006/09/sip/roaming-self** document.

The server MUST send back a 400 Bad Request response if the delegator URI is identical to the delegate **uri** specified in the request.

The server SHOULD send back a 403 Forbidden response if the **From-URI** and the **To-URI** are not identical.<48>

The server MUST send back a 403 Forbidden response if the action is "remove" and the delegate is not found in the **delegateSet** of the delegator.

The server (2) MUST send back a 404 Not Found if the delegate specified in the request is not a valid URI in this deployment.

The server MUST send back a 409 Conflict response if the version check fails. The format of this message is specified in section 2.2.2.10.4.

The server MUST send back a 413 Request Entity Too Large response if the request is being denied because it caused the delegate quota to be exceeded.

3.9.6 Timer Events

None.

3.9.7 Other Local Events

None.

3.10 Unified Contact Store Details

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.

An enhanced presence server maintains the elements for the groups and contact list as described in [MS-SIP] section 3.7.<49>

3.10.1 Abstract Data Model

None.

3.10.2 Timers

None.

3.10.3 Initialization

None.

3.10.4 Higher-Layer Triggered Events

- **Start Migration:** When the enhanced presence server receives a subscribe request for the vnd-microsoft-roaming-contact with the supported header of ms-ucs-ready it add the user to the queue of users to whose contacts need to be migrated to the email server. When the server chooses to migrate the contacts of the user is an implementation detail.
- **During Migration:** The contact and group list is put in a read only mode. All SERVICE messages meant to modify the contact list during this point will be rejected by the enhanced presence server (2).
- **Post Migration:** SIP enabled Endpoints that would like the contact and group list from the enhanced presence Server MUST NOT put the ms-ucs header in the subscribe request. The enhanced presence server will then proxy all changes to the endpoint. Whether the server decided to send delta updates or the entire contact and group list is implementation specific. Whether the server chooses to proxy only reads or reads and writes to the email server contact store is also an implementation detail.

SIP enabled endpoints that can connect to the email server SHOULD add the ms-ucs supported header in the subscribe request that is sent to the enhanced presence server.

3.10.4.1 Error Responses

Apart from the error codes sent by the enhanced presence server (2) detailed in [MS-SIP] section 3.7 the error codes detailed in [MS-OCER] section 7.3 will be sent.

3.10.5 Timer Events

None.

3.10.6 Other Local Events

None.

4 Protocol Examples

4.1 Directory Search

In the following example, the user bob@contoso.com is issuing a DS user search against the SIP domain contoso.com and searching for all users whose name begins with "Alice".

```
SERVICE sip:contoso.com SIP/2.0
Via: SIP/2.0/TLS 192.168.66.91:49541
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=0d18c6dde0;epid=0199305c6d
To: <sip:contoso.com>
Call-ID: 717ff9c645de415688683fb482527aa0
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:E4abE0hR5VCdvhGcA7shLQAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/SOAP+xml
Content-Length: …

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <m:directorySearch xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip">
      <m:filter m:href="#searchArray"/>
      <m:maxResults>100</m:maxResults>
    </m:directorySearch>
    <m:Array xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip" m:id="searchArray">
      <m:row m:attrib="givenName" m:value="Alice"/>
    </m:Array>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The search results for the given name Alice.

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: <sip:bob@contoso.com>;tag=0d18c6dde0;epid=0199305c6d
To: <sip:contoso.com>;tag=32317843A7E3CC862EE01750D91100E1
Call-ID: 717ff9c645de415688683fb482527aa0
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 192.168.66.91:49541;ms-received-port=49541;ms-received-cid=554400
Content-Type: application/SOAP+xml

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip">
  <SOAP-ENV:Body>
    <m:directorySearch>
      <m:moreAvailable>false</m:moreAvailable>
      <m:returned>1</m:returned>
      <m:value m:href="#rows"/>
    </m:directorySearch>
    <m:Array m:id="rows">
      <m:row m:uri="sip:Alice@contoso.com" m:displayName="Alice"
             m:title="SR. LEAD" m:office="111" m:phone="+1 (555) 555555 X55555"
             m:company="contoso" m:city="" m:state="" m:country=""
             m:email="Alice@contoso.com"/>
    </m:Array>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
.
*
```

The following example is a fault response to a DS user search

```
SIP/2.0 400 Bad request
Authentication-Info: …
Content-Length: …
From: <sip:bob@contoso.com>;tag=0d18c6dde0;epid=0199305c6d
To: <sip:contoso.com>;tag=32317843A7E3CC862EE01750D91100E1
Call-ID: 717ff9c645de415688683fb482527aa0
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 10.56.66.91:49541;ms-received-port=49541;ms-received-cid=554400
Content-Type: application/SOAP+xml

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" >
  <SOAP-ENV:Body>
    <SOAP-EVN:Fault>
      <Faultcode>Client.BadCall</Faultcode>    </SOAP-ENF:Fault>
  </SOAP-ENV:Body>


</SOAP-ENV:Envelope>
```

4.2 Publish

4.2.1 Publishing a Category Instance

The following example shows a **publish** request by bob@contoso.com to publish an instance of the **note category** to containers 200, 300, and 400.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=b5410171e2;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 82369a8c95ba4778b3b9220e4abb73d8
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-category-publish+xml
Content-Length: …

<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="note" instance="0" container="300" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
    <publication categoryName="note" instance="0" container="400" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
    <publication categoryName="note" instance="0" container="200" version="0"
expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </publication>
  </publications>
</publish>
```

The next example shows a server sending a 200 OK response to bob@contoso.com. The response contains an embedded **categories** document.

```
SIP/2.0 200 OK
Authentication-Info: …
From: "Bob"<sip:bob@contoso.com>;tag=b5410171e2;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 82369a8c95ba4778b3b9220e4abb73d8
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms-received-cid=B00
Content-Type: application/vnd-microsoft-roaming-self+xml

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="200" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="300" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="400" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
  </categories>
</roamingData>
```

The server sends a BENOTIFY message for all active self subscription dialogs of bob@contoso.com.

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK45F0BC36.F559A04F;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 15 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46487
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="200" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
```

```
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="300" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:29:13.263"
container="400" version="1" expireType="static">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
        <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </category>
  </categories>
</roamingData>
```

The next example shows a 409 Conflict response that is sent in place of the previous 200 OK response if a **category** instance version in a publication request is not equal to the server version of that **category** instance. The body of a 409 Conflict response contains information about the version number specified in the request and the current version number at the server. When a server sends a 409 Conflict response to a publisher, a BENOTIFY message is not sent to the subscribers of the publisher.

The following example illustrates the format of such a message:

```
SIP/2.0 409 Conflict
Authentication-Info: …
Content-Length: …
From: "bob"<sip:bob@contoso.com>;tag=47814a8299;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 82369a8c95ba4778b3b9220e4abb73d8
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 172.24.32.124:62493;ms-received-port=62493;ms-received-cid=7E00
ms-diagnostics: 2044;reason="Publication version out of date";source="..."
Content-Type: application/msrtc-fault+xml

<Fault>
  <Faultcode>Protocol client.BadCall.WrongDelta</Faultcode>
  <details>
    <operation index="1" version="1" curVersion="2">
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
         <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </operation>
    <operation index="2" version="1" curVersion="2" >
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
         <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </operation>
    <operation index="3" version="1" curVersion="2" >
      <note xmlns="http://schemas.microsoft.com/2006/09/sip/note">
         <body type="personal" uri="">Working until 5pm today</body>
      </note>
    </operation>
  </details>
</Fault>
```

4.2.2 Clearing a Category Instance

The following example shows a **publish** request by bob@contoso.com to clear an instance of the **note category** from containers 200, 300, and 400.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:54059
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=f311784039;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 9c41452c43ee4cef815d6a61906fc755
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-category-publish+xml
Content-Length: …
<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="note" instance="0" container="300" version="1"
expireType="static" expires="0" />
    <publication categoryName="note" instance="0" container="200" version="1"
expireType="static" expires="0" />
    <publication categoryName="note" instance="0" container="400" version="1"
expireType="static" expires="0" />
  </publications>
</publish>
```

The next example shows a 200 OK response from the server to bob@contoso.com that contains an embedded categories document.

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>;tag=f311784039;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 9c41452c43ee4cef815d6a61906fc755
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:54059;ms-received-port=54059;ms-received-cid=D00
Content-Type: application/vnd-microsoft-roaming-self+xml
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
            xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
            xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
            xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="note" container="200" />
    <category name="note" container="300" />
    <category name="note" container="400" />
  </categories>
</roamingData>
```

The preceding example results in a BENOTIFY being sent for all active self subscription dialogs.

```
BENOTIFY sip:172.24.32.124:54059;transport=tcp;ms-opaque=3a7258e2aa;ms-received-cid=D00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK429E5FBE.AF03749F;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=ea035dd987;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=0C390080
Call-ID: fb6533aac0034dd2a912d05d3f238a25
CSeq: 5 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=51920
```

```
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="note" container="200" />
    <category name="note" container="300" />
    <category name="note" container="400" />
  </categories>
</roamingData>
```

4.3 Aggregation

4.3.1 Aggregation of State Category

This example demonstrates the publication of a **userState** instance with availability in the **DoNotDisturb** class.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=f99fe8ef36;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: dcd296aba16c4f10921e971d20998bbf
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: Kerberos qop="auth", realm="SIP Communications Service",
opaque="091C4306", crand="35e20c02", cnum="35", targetname="sip/server.contoso.com",
response="602306092a864886f71201020201011100ffffffff10067a4f2fbba68318a4437e5586ff8a"
Content-Type: application/msrtc-category-publish+xml
Content-Length: …

<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="state" instance="603979776" container="2" version="0"
expireType="time" expires="0" />
    <publication categoryName="state" instance="536870912" container="3" version="0"
expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state" manual="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="userState">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" minAvailability="6900"
maxAvailability="8999" />
        <endpointLocation>
        </endpointLocation>
      </state>
    </publication>
    <publication categoryName="state" instance="536870912" container="2" version="0"
expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state" manual="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="userState">
        <availability>9500</availability>
        <activity minAvailability="9500" maxAvailability="11999">
          <custom LCID="1033">Interviewing</custom>
        </activity>
        <endpointLocation>
        </endpointLocation>
      </state>
    </publication>
    <publication categoryName="state" instance="603979776" container="3" version="0"
expireType="time" expires="0" />
  </publications>
```

```
</publish>
```

In the next example, the server sends a 200 OK response that contains aggregated **state category** instances.

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>;tag=f311784039;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 9c41452c43ee4cef815d6a61906fc755
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:54059;ms-received-port=54059;ms-received-cid=D00
Content-Type: application/vnd-microsoft-roaming-self+xml
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="2" version="6" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
        <activity>
          <custom LCID="1033"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">Interviewing</custom>
        </activity>
        <endpointLocation>Home Custom EndPoint Location</endpointLocation>
    <delimiter
    xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>      </state>
    </category>
    <category name="state" instance="268435456" publishTime="2008-01-11T17:23:10.030"
container="2" version="1" expireType="user">
      <state xsi:type="aggregateMachineState" endpointId="221ef77e-3a68-5570-86ed-
6ea5bd4b7ff8" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="state" instance="809938687" publishTime="2008-01-11T17:25:19.233"
container="2" version="2" expireType="endpoint" endpointId="221EF77E-3A68-5570-86ED-
6EA5BD4B7FF8">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
      </state>
    <delimiter
    xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>    </category>
    <category name="state" instance="536870912" publishTime="2008-01-11T17:28:24.217"
container="2" version="1" expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>9500</availability>
```

```
        <activity minAvailability="9500" maxAvailability="11999">
          <custom LCID="1033">Interviewing</custom>
        </activity>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="3" version="6" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" />
        <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
    <delimiter
    xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>      </state>
    </category>
    <category name="state" instance="809938687" publishTime="2008-01-11T17:25:19.233"
container="3" version="2" expireType="endpoint" endpointId="221EF77E-3A68-5570-86ED-
6EA5BD4B7FF8">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
    <delimiter
    xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>      </state>
    </category>
    <category name="state" instance="536870912" publishTime="2008-01-11T17:28:24.217"
container="3" version="1" expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" minAvailability="6900"
maxAvailability="8999">
        </activity>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="100" version="6" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2008-01-11T17:28:24.217"
container="100" version="6" expireType="user">
      <legacyInterop availability="9500" />
    </category>
    <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="200" version="6" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
        <activity>
          <custom LCID="1033"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">Interviewing</custom>
```

```
        </activity>
      <delimiter
      xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
       <device>computer</device>
      <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
        </state>
      </category>
      <category name="legacyInterop" instance="1" publishTime="2008-01-11T17:28:24.217"
container="200" version="6" expireType="user">
        <legacyInterop availability="9500" />
      </category>
      <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="300" version="6" expireType="user">
        <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
          <availability>6900</availability>
          <activity token="urgent-interruptions-only" />
          <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
      <delimiter
      xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
       <timeZoneBias>999</timeZoneBias>
       <timeZoneName>Pacific Daylight Time</timeZoneName>
       <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
       <device>computer</device>
      <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>      </state>
      </category>
      <category name="legacyInterop" instance="1" publishTime="2008-01-11T17:28:24.217"
container="300" version="6" expireType="user">
        <legacyInterop availability="6900" token="urgent-interruptions-only" />
      </category>
      <category name="state" instance="1" publishTime="2008-01-11T17:28:24.217"
container="400" version="6" expireType="user">
        <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
          <availability>9500</availability>
          <activity>
            <custom LCID="1033"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">Interviewing</custom>
          </activity>
          <endpointLocation>Home Custom EndPoint Location</endpointLocation>
      <delimiter
      xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
       <timeZoneBias>999</timeZoneBias>
       <timeZoneName>Pacific Daylight Time</timeZoneName>
       <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
       <device>computer</device>
      <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>      </state>
      </category>
      <category name="legacyInterop" instance="1" publishTime="2008-01-11T17:28:24.217"
container="400" version="6" expireType="user">
        <legacyInterop availability="9500" />
      </category>
    </categories>
  </roamingData>
```

4.3.1.1 State Category Aggregation Walkthrough

The following example shows state instances in container 2:

```
  <category name="state" instance="603979776"
             publishTime="2008-01-11T18:26:06.450"
             container="2" version="1"
             expireType="time" expires="800">
   <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          manual="true" xsi:type="userState">
```

```
    <availability>9000</availability>
    <endpointLocation>
    </endpointLocation>
  </state>
</category>

<category name="state" instance="809938687"
          publishTime="2008-01-11T19:06:03.920"
          container="2" version="9" expireType="endpoint"
          endpointId="221EF77E-3A68-5570-86ED-6EA5BD4B7FF8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" xsi:type="machineState">
    <availability>5000</availability>
    <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
  </state>
</category>

<category name="state" instance="1339299275"
          publishTime="2008-01-11T19:06:28.937"
          container="2" version="1" expireType="endpoint"
          endpointId="221EF77E-3A68-5570-86ED-6EA5BD4B7FF8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" uri="john@contoso.com"
         startTime="2008-01-11T19:00:00Z" xsi:type="calendarState">
    <availability>6500</availability>
    <activity token="in-a-meeting" minAvailability="6500" maxAvailability="8999">
    </activity>
    <endpointLocation>
    </endpointLocation>
    <meetingSubject>Customer Meeting</meetingSubject>
    <meetingLocation>Conf Room 100</meetingLocation>
  </state>
</category>
```

The following example shows state instances in container 3:

```
<category name="state" instance="603979776"
          publishTime="2008-01-11T18:26:06.450" container="3"
          version="1" expireType="time" expires="800">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="true" xsi:type="userState">
    <availability>6900</availability>
    <activity token="urgent-interruptions-only"
              minAvailability="6900" maxAvailability="8999">
    </activity>
    <endpointLocation>
    </endpointLocation>
  </state>
</category>

<category name="state" instance="809938687"
          publishTime="2008-01-11T19:06:03.920"
          container="3" version="9" expireType="endpoint"
          endpointId="221EF77E-3A68-5570-86ED-6EA5BD4B7FF8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" xsi:type="machineState">
    <availability>5000</availability>
    <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
```

```
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  </state>
</category>

<category name="state" instance="1339299275"
          publishTime="2008-01-11T19:06:28.937"
          container="3" version="1" expireType="endpoint"
          endpointId="221EF77E-3A68-5570-86ED-6EA5BD4B7FF8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" uri="john@contoso.com"
         startTime="2008-01-11T19:00:00Z" xsi:type="calendarState">
    <availability>6500</availability>
    <activity token="in-a-meeting" minAvailability="6500" maxAvailability="8999">
    </activity>
    <endpointLocation>
    </endpointLocation>
    <meetingSubject>Customer Meeting</meetingSubject>
    <meetingLocation>Conf Room 100</meetingLocation>
  </state>
</category>
```

The following example shows state instances produced by the computing function:

```
<category name="state" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="2" version="11" expireType="user">
  <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>9000</availability>
    <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
    <meetingSubject>Customer Meeting</meetingSubject>
    <meetingLocation>Conf Room 100</meetingLocation>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  </state>
</category>

<category name="state" instance="268435456"
          publishTime="2008-01-11T19:06:03.920"
          container="2" version="9" expireType="user">
  <state xsi:type="aggregateMachineState"
         endpointId="221ef77e-3a68-5570-86ed-6ea5bd4b7ff8"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>5000</availability>
  </state>
</category>

<category name="state" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="100" version="11" expireType="user">
  <state xsi:type="aggregateState"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>9000</availability>
  </state>
</category>

<category name="legacyInterop" instance="1"
          publishTime="2008-01-11T19:06:28.937"
```

```
          container="100" version="11" expireType="user">
  <legacyInterop availability="9000" />
</category>

<category name="state" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="200" version="11" expireType="user">
  <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>9000</availability>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  </state>
</category>

<category name="legacyInterop" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="200" version="11" expireType="user">
  <legacyInterop availability="9000" />
</category>

<category name="state" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="400" version="11" expireType="user">
  <state xsi:type="aggregateState"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>9000</availability>
    <endpointLocation>Work Custom Endpoint Location</endpointLocation>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  </state>
</category>

<category name="legacyInterop" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="400" version="11" expireType="user">
  <legacyInterop availability="9000" />
</category>

<category name="state" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="3" version="15" expireType="user">
  <state xsi:type="aggregateState" lastActive="2008-01-11T19:06:03"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>8400</availability>
    <activity token="urgent-interruptions-only" />
    <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
    <meetingSubject>Customer Meeting</meetingSubject>
    <meetingLocation>Conf Room 100</meetingLocation>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
</state>
</category>

<category name="state" instance="1"
```

```
          publishTime="2008-01-11T19:06:28.937"
          container="300" version="15" expireType="user">
  <state xsi:type="aggregateState"
         lastActive="2008-01-11T19:06:03"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>8400</availability>
    <activity token="urgent-interruptions-only" />
    <endpointLocation>Work_Custom_Endpoint_Location</endpointLocation>
    <meetingSubject>Customer Meeting</meetingSubject>
    <meetingLocation>Conf Room 100</meetingLocation>
    <delimiter
     xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
     <timeZoneBias>999</timeZoneBias>
     <timeZoneName>Pacific Daylight Time</timeZoneName>
     <timeZoneAbbreviation>PDT</timeZoneAbbreviation>
     <device>computer</device>
    <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes"/>
  </state>
</category>

<category name="legacyInterop" instance="1"
          publishTime="2008-01-11T19:06:28.937"
          container="300" version="15" expireType="user">
  <legacyInterop availability="8400" token="urgent-interruptions-only" />
</category>
```

4.3.2 Aggregation of Device Category

In the following example, **device** and **state category** instances are published to container 2:

```
SERVICE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.32.208:1996
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=26b7c6e008;epid=2be779d608
To: <sip:alice@contoso.com>
Call-ID: 2ec1255d7a6c4ab38ff0d979eb2e340c
CSeq: 1 SERVICE
Contact: <sip:alice@contoso.com;opaque=user:epid:xwcovZCFDlO1TJPHSfLkcAAA;gruu>
User-Agent: UCCP/2.0.6362.36 OC/2.0.6362.36 (Microsoft Office Communicator)
Proxy-Authorization: ...
Content-Type: application/msrtc-category-publish+xml
Content-Length: …
<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:alice@contoso.com">
    <publication categoryName="device" instance="3790621588" container="2" version="0"
expireType="endpoint">
      <device xmlns="http://schemas.microsoft.com/2006/09/sip/device" endpointId="BD2807C7-
8590-530E-B54C-93C749F2E470">
        <capabilities preferred="false" uri="sip:alice@contoso.com">
          <text capture="true" render="true" publish="false" />
          <gifInk capture="false" render="true" publish="false" />
          <isfInk capture="false" render="true" publish="false" />
          <breakthrough render="false" capture="false" publish="true"/>
          <applicationSharing render="true" capture="true"
            publish="false"/>
        </capabilities>
        <capabilities preferred="false" uri="Alice@contoso.com">
          <calendar capture="false" render="false" publish="true" version="655616" />
        </capabilities>
        <timezone>00:00:00-00:00:00</timezone>
        <machineName>ALICE-MACHINE12</machineName>
      </device>
    </publication>
    <publication categoryName="state" instance="1042220217" container="2" version="0"
expireType="endpoint">
```

```
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state" manual="false"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </publication>
    <publication categoryName="state" instance="1042220217" container="3" version="0"
expireType="endpoint">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state" manual="false"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </publication>
  </publications>
</publish>
```

The server sends a 200 OK response that contains **aggregateState** and **services category** instances, as follows:

```
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:alice@contoso.com">
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950" container="2"
version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="state" instance="268435456" publishTime="2008-01-24T19:45:38.950"
container="2" version="20" expireType="user">
      <state xsi:type="aggregateMachineState" endpointId="bd2807c7-8590-530e-b54c-
93c749f2e470" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="state" instance="967115030" publishTime="2008-01-24T19:44:20.873"
container="2" version="4" expireType="endpoint" endpointId="6F2B83F8-01A1-5BFC-A93A-
FC855C8E0754">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>5000</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="1042220217" publishTime="2008-01-24T19:45:38.950"
container="2" version="1" expireType="endpoint" endpointId="BD2807C7-8590-530E-B54C-
93C749F2E470">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="536870912" publishTime="2008-01-24T19:01:35.903"
container="2" version="2" expireType="static">
```

```
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>6500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="device" instance="2588938605" publishTime="2008-01-24T19:15:43.999"
container="2" version="1" expireType="endpoint" endpointId="6F2B83F8-01A1-5BFC-A93A-
FC855C8E0754">
      <device xmlns="http://schemas.microsoft.com/2006/09/sip/device" endpointId="6F2B83F8-
01A1-5BFC-A93A-FC855C8E0754">
        <capabilities preferred="false" uri="sip:alice@contoso.com">
          <text capture="true" render="true" publish="false">
          </text>
          <gifInk capture="false" render="true" publish="false">
          </gifInk>
          <isfInk capture="false" render="true" publish="false">
          </isfInk>
          <breakthrough render="false" capture="false" publish="true">
          </breakthrough>
          <applicationSharing render="true" capture="true"
            publish="false">
          </applicationSharing>
        </capabilities>
        <timezone>00:00:00-00:00</timezone>
        <machineName>ALICE-MACHINE11</machineName>
      </device>
    </category>
    <category name="device" instance="3790621588" publishTime="2008-01-24T19:45:38.950"
container="2" version="1" expireType="endpoint" endpointId="BD2807C7-8590-530E-B54C-
93C749F2E470">
      <device xmlns="http://schemas.microsoft.com/2006/09/sip/device" endpointId="BD2807C7-
8590-530E-B54C-93C749F2E470">
        <capabilities preferred="false" uri="sip:alice@contoso.com">
          <text capture="true" render="true" publish="false">
          </text>
          <gifInk capture="false" render="true" publish="false">
          </gifInk>
          <isfInk capture="false" render="true" publish="false">
          </isfInk>
          <breakthrough render="false" capture="false" publish="true">
          </breakthrough>
          <applicationSharing render="true" capture="true"
            publish="false">
          </applicationSharing>
        </capabilities>
        <capabilities preferred="false" uri="Alice@contoso.com">
          <calendar capture="false" render="false" publish="true" version="655616">
          </calendar>
        </capabilities>
        <timezone>00:00:00-00:00</timezone>
        <machineName>ALICE-MACHINE12</machineName>
      </device>
    </category>
    <category name="services" instance="0" publishTime="2008-01-24T19:45:38.950"
container="2" version="21" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" publish="false"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
            <gifInk render="true" capture="false" publish="false"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
            <isfInk render="true" capture="false" publish="false"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
            <breakthrough render="false" capture="false" publish="true"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
```

```
            <applicationSharing render="true" capture="true" publish="false"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
          </capabilities>
        </service>
        <service uri="Alice@contoso.com">
          <capabilities>
            <calendar render="false" capture="false" publish="true" version="655616"
preferredEndpointId="bd2807c7-8590-530e-b54c-93c749f2e470" deviceAvailability="3500" />
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950" container="3"
version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="state" instance="967115030" publishTime="2008-01-24T19:44:20.873"
container="3" version="4" expireType="endpoint" endpointId="6F2B83F8-01A1-5BFC-A93A-
FC855C8E0754">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>5000</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="1042220217" publishTime="2008-01-24T19:45:38.950"
container="3" version="1" expireType="endpoint" endpointId="BD2807C7-8590-530E-B54C-
93C749F2E470">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="536870912" publishTime="2008-01-24T19:01:35.903"
container="3" version="2" expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>6500</availability>
        <endpointLocation>
        </endpointLocation>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950"
container="100" version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2008-01-24T19:45:38.950"
container="100" version="12" expireType="user">
      <legacyInterop availability="6500" />
    </category>
    <category name="services" instance="0" publishTime="2008-01-24T19:45:38.950"
container="100" version="10" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500" />
            <gifInk render="true" capture="false" deviceAvailability="3500" />
            <isfInk render="true" capture="false" deviceAvailability="3500" />
```

```
            <applicationSharing render="true" capture="true"
               deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950"
container="200" version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2008-01-24T19:45:38.950"
container="200" version="12" expireType="user">
      <legacyInterop availability="6500" />
    </category>
    <category name="services" instance="0" publishTime="2008-01-24T19:45:38.950"
container="200" version="10" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500" />
            <gifInk render="true" capture="false" deviceAvailability="3500" />
            <isfInk render="true" capture="false" deviceAvailability="3500" />
            <applicationSharing render="true" capture="true"
              deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950"
container="300" version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2008-01-24T19:45:38.950"
container="300" version="12" expireType="user">
      <legacyInterop availability="6500" />
    </category>
    <category name="services" instance="0" publishTime="2008-01-24T19:45:38.950"
container="300" version="10" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500" />
            <gifInk render="true" capture="false" deviceAvailability="3500" />
            <isfInk render="true" capture="false" deviceAvailability="3500" />
            <applicationSharing render="true" capture="true"
              deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1" publishTime="2008-01-24T19:45:38.950"
container="400" version="12" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2008-01-24T19:45:38.950"
container="400" version="12" expireType="user">
      <legacyInterop availability="6500" />
    </category>
    <category name="services" instance="0" publishTime="2008-01-24T19:45:38.950"
container="400" version="10" expireType="user">
```

```
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500" />
            <gifInk render="true" capture="false" deviceAvailability="3500" />
            <isfInk render="true" capture="false" deviceAvailability="3500" />
            <applicationSharing render="true" capture="true"
              deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
  </categories>
</roamingData>
```

4.3.2.1 Device Category Aggregation

In this example, **Endpoint1** for Bob has the following instances:

```
<category name="device" instance="2210988456"
          publishTime="2008-01-22T22:14:36.780" container="2" version="2"
          expireType="endpoint"
          endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
  <device xmlns="http://schemas.microsoft.com/2006/09/sip/device"
          endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">

    <capabilities preferred="false" uri="sip:bob@contoso.com">
      <text capture="true" render="true" publish="false">
      </text>
      <gifInk capture="false" render="true" publish="false">
      </gifInk>
      <isfInk capture="false" render="true" publish="false">
      </isfInk>
    </capabilities>
    <capabilities preferred="false" uri="tel:+11234567890;ext=67890">
      <remoteCallControl capture="false" render="false" publish="true">
      </remoteCallControl>
    </capabilities>
    <timezone>00:00:00-00:00</timezone>
    <machineName>BOB-1</machineName>
  </device>
</category>

<category name="state" instance="943493146"
          publishTime="2008-01-22T22:14:36.233" container="2" version="1"
          expireType="endpoint"
          endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" xsi:type="machineState">
    <availability>3500</availability>
    <endpointLocation>
    </endpointLocation>
  </state>
</category>
```

Endpoint2 for Bob has the following instances:

```
<category name="device" instance="1827441964"
          publishTime="2008-01-22T22:20:22.313"
          container="2" version="1" expireType="endpoint"
          endpointId="2CD4F7CA-B1D1-5EDA-8D79-79EE4298414D">
  <device xmlns="http://schemas.microsoft.com/2006/09/sip/device"
          endpointId="2CD4F7CA-B1D1-5EDA-8D79-79EE4298414D">
```

```
    <capabilities preferred="false" uri="sip:bob@contoso.com">
      <text capture="true" render="true" publish="false">
      </text>
      <gifInk capture="false" render="true" publish="false">
      </gifInk>
      <isfInk capture="false" render="true" publish="false">
      </isfInk>
    </capabilities>
    <capabilities preferred="false" uri="bob@exchange.contoso.com">
      <calendar capture="false" render="false" publish="true" version="655616" />
    </capabilities>
    <timezone>00:00:00-00:00</timezone>
    <machineName>BOB-2</machineName>
  </device>
</category>

<category name="state" instance="919521490"
          publishTime="2008-01-22T22:20:22.313" container="2" version="1"
          expireType="endpoint"
          endpointId="2CD4F7CA-B1D1-5EDA-8D79-79EE4298414D">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false"
         xsi:type="machineState">
    <availability>3500</availability>
    <endpointLocation>
    </endpointLocation>
  </state>
</category>
```

The computing function produces the following URI set:

```
sip:bob@contoso.com
bob@exchange.contoso.com
tel:+11234567890;ext=67890
```

The computing function produces **service** elements for the following URIs:

```
<service uri="sip:bob@contoso.com">
  <capabilities>
  </capabilities>
</service>
<service uri="tel:+11234567890;ext=67890">
  <capabilities>
  </capabilities>
</service>
<service uri="bob@exchange.contoso.com">
  <capabilities>
  </capabilities>
</service>
```

The computing function produces the following **services** instance for self, as follows:

```
<category name="services" instance="0" publishTime="2008-01-22T22:20:22.313"
          container="2" version="3" expireType="user">
  <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">

    <service uri="sip:bob@contoso.com">
      <capabilities>
        <text render="true" capture="true" publish="false"
              preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
              deviceAvailability="3500" />
        <gifInk render="true" capture="false" publish="false"
                preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
```

```
                deviceAvailability="3500" />
          <isfInk render="true" capture="false" publish="false"
                  preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                  deviceAvailability="3500" />
        </capabilities>
      </service>

      <service uri="tel:+11234567890;ext=67890">
        <capabilities>
          <remoteCallControl render="false" capture="false" publish="true"
                             preferredEndpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
                             deviceAvailability="3500" />
        </capabilities>
      </service>

      <service uri="bob@exchange.contoso.com">
        <capabilities>
          <calendar render="false" capture="false" publish="true"
                    version="655616"
                    preferredEndpointId="2cd4f7ca-b1d1-5eda-8d79-79ee4298414d"
                    deviceAvailability="3500" />
        </capabilities>
      </service>
    </services>

  </category>
```

The computing function produces the following **services** instance for subscribers:

```
  <category name="services" instance="0" publishTime="2008-01-22T22:20:22.313"
            container="100" version="3" expireType="user">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">

      <service uri="sip:bob@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500" />
          <gifInk render="true" capture="false" deviceAvailability="3500" />
          <isfInk render="true" capture="false"  deviceAvailability="3500" />
        </capabilities>
      </service>
    </services>

  </category>
```

4.3.3 Creation of workingHours Category

In this example, **calendarData category** instances are published into containers 1, 100, 200, 300, 400 and 32000.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.32.214:52681
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=9e49e11dfc;epid=3b09add8bc
To: <sip:bob@contoso.com>
Call-ID: 0f34c7038250411e98faf053a039b89b
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:Q_zZt9cZEVOeAMbFsdgb-wAA;gruu>
User-Agent: UCCAPI/4.0.7254.0 OC/4.0.7254.0 (Microsoft Communicator 2010 (Beta))
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="6B4B0A23", targetname="TK5UCDFPL01F02.exchange.corp.microsoft.com", crand="7ecb3404",
cnum="10", response="397f685dd70c1b412a8045ef29d3e82c255182c7"
Content-Type: application/msrtc-category-publish+xml
Content-Length: ...
```

```
<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="calendarData" instance="1982801780" container="200"
version="0" expireType="endpoint">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="1982801780" container="1" version="0"
expireType="endpoint">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="0" container="400" version="1"
expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <WorkingHours xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="1982801780" container="300"
version="0" expireType="endpoint">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="0" container="300" version="1"
expireType="static">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
```

```
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="1982801780" container="400"
version="0" expireType="endpoint">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="1982801780" container="100"
version="0" expireType="endpoint">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData" />
    </publication>
    <publication categoryName="calendarData" instance="0" container="32000" version="1"
expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData" />
    </publication>
    <publication categoryName="calendarData" instance="0" container="100" version="1"
expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData" />
    </publication>
    <publication categoryName="calendarData" instance="1982801780" container="32000"
version="0" expireType="endpoint">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData" />
    </publication>
    <publication categoryName="calendarData" instance="0" container="200" version="1"
expireType="static">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
```

```
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </publication>
    <publication categoryName="calendarData" instance="0" container="1" version="1"
expireType="static">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </publication>
  </publications>
</publish>
```

In the next example, the server sends a 200 OK response that contains **workingHours** and **calendarData category** instances. The computing function in the server creates a **workingHours category** instance from the **calendarData category** (instance zero) in container 1.

```
SIP/2.0 200 OK
Authentication-Info: TLS-DSK qop="auth", opaque="6B4B0A23", srand="5A631FE0", snum="10",
rspauth="00090be99d78b9ec33004adce8e6e8d96d22ff07",
targetname="TK5UCDFPL01F02.exchange.corp.microsoft.com", realm="SIP Communications Service",
version=4
Content-Length: 22814
From: "Bob"<sip:bob@contoso.com>;tag=9e49e11dfc;epid=3b09add8bc
To: <sip:bob@contoso.com>;tag=97C972C4F65C13BB1CEE363598650D9A
Call-ID: 0f34c7038250411e98faf053a039b89b
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 172.24.32.214:52681;received=157.54.78.92;ms-received-port=52681;ms-
received-cid=140E5C00
Content-Type: application/vnd-microsoft-roaming-self+xml
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
```

```
xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="1" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
      </calendarData>
    </category>
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="1" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </category>
    <category name="workingHours" instance="0" publishTime="2010-02-15T12:31:02.813"
container="1" version="2" expireType="static">
      <calendarData mailboxID="bob@contoso.com"
xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
```

```
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </category>
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="100" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
      </calendarData>
    </category>
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="100" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
      </calendarData>
    </category>
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="200" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </category>
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="200" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </category>
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="300" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </category>
```

```
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="300" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <WorkingHours xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/calendarData"
xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </category>
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="400" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <freeBusy startTime="2010-02-14T08:00:00Z" granularity="PT15M" encodingVersion="1">
        </freeBusy>
      </calendarData>
    </category>
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="400" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="bob@contoso.com">
        <WorkingHours xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
          <TimeZone>
            <Bias>480</Bias>
            <StandardTime>
              <Bias>0</Bias>
              <Time>02:00:00</Time>
              <DayOrder>1</DayOrder>
              <Month>11</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </StandardTime>
            <DaylightTime>
              <Bias>-60</Bias>
              <Time>02:00:00</Time>
              <DayOrder>2</DayOrder>
              <Month>3</Month>
              <DayOfWeek>Sunday</DayOfWeek>
            </DaylightTime>
          </TimeZone>
          <WorkingPeriodArray>
            <WorkingPeriod>
              <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
              <StartTimeInMinutes>540</StartTimeInMinutes>
              <EndTimeInMinutes>1020</EndTimeInMinutes>
```

```
            </WorkingPeriod>
          </WorkingPeriodArray>
        </WorkingHours>
      </calendarData>
    </category>
    <category name="calendarData" instance="1982801780" publishTime="2010-02-15T12:31:02.813"
container="32000" version="1" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
      </calendarData>
    </category>
    <category name="calendarData" instance="0" publishTime="2010-02-15T12:31:02.813"
container="32000" version="2" expireType="static">
      <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData">
      </calendarData>
    </category>
  </categories>
</roamingData>
```

4.3.4 Creation of dndState Category

In this example, **state category** instances of xsi:type userState are published into containers 2 and 3.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.32.214:52777
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=c9a99869ef;epid=3b09add8bc
To: <sip:bob@contoso.com>
Call-ID: 9d28ef15c5314a4f8ce09c44aa0d574c
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:Q_zZt9cZEVOeAMbFsdgb-wAA;gruu>
User-Agent: UCCAPI/4.0.7254.0 OC/4.0.7254.0 (Microsoft Communicator 2010 (Beta))
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="9742C9CC", targetname="Server.contoso.com", crand="6f8cafb1", cnum="468",
response="a8e9cfe8f49d872d611670bac6b6408d16ab0a06"
Content-Type: application/msrtc-category-publish+xml
Content-Length: …

<publish xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence">
  <publications uri="sip:bob@contoso.com">
    <publication categoryName="state" instance="536870912" container="3"
        version="0" expireType="static" expires="0" />
    <publication categoryName="state" instance="536870912" container="2"
        version="0" expireType="static" expires="0" />
    <publication categoryName="state" instance="603979776" container="3"
        version="0" expireType="time" expires="86400">
      <state manual="true" xsi:type="userState"
          xmlns="http://schemas.microsoft.com/2006/09/sip/state"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" minAvailability="6900"
           maxAvailability="8999" />
      </state>
    </publication>
    <publication categoryName="state" instance="603979776" container="2"
        version="0" expireType="time" expires="86400">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state" manual="true"


            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="userState">


        <availability>9000</availability>
      </state>
    </publication>
```

```
    </publications>
  </publish>
```

In the next example, the server (2) sends a 200 OK response that contains the **dndState** and **state category** instances. The computing function in the server creates and publishes **dndState category** instances based on the **userState** instance of the **state category**.

```
SIP/2.0 200 OK
Authentication-Info: TLS-DSK qop="auth", opaque="9742C9CC", srand="E87BD197", snum="599",
rspauth="148204fdaeee167c532fe061d3ecfe02b52efe86", targetname="Server.contoso.com",
realm="SIP Communications Service", version=4
Content-Length: 9757
From: "Bob"<sip:bob@contoso.com>;tag=c9a99869ef;epid=3b09add8bc
To: <sip:bob@contoso.com>;tag=97C972C4F65C13BB1CEE363598650D9A
Call-ID: 9d28ef15c5314a4f8ce09c44aa0d574c
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 172.24.32.214:52777;received=157.54.78.92;ms-received-port=52777;ms-
received-cid=FB6DE00
Content-Type: application/vnd-microsoft-roaming-self+xml

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
  uri="sip:bob@contoso.com">
    <category name="dndState" instance="0" publishTime="2010-02-13T09:01:58.740"
container="0" version="6" expireType="static">
      <state xsi:type="userState" manual="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="2"
version="559" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9000</availability>
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
      </state>
    </category>
    <category name="state" instance="268435456" publishTime="2010-02-13T08:40:58.693"
container="2" version="513" expireType="user">
      <state xsi:type="aggregateMachineState" endpointId="b7d9fc43-19d7-5311-9e00-
c6c5b1d81bfb" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="state" instance="816379399" publishTime="2010-02-13T00:04:03.067"
container="2" version="102" expireType="endpoint" endpointId="5D4FEF3F-71C1-5562-A3D9-
56746B756BCB">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false" xsi:type="machineState">
        <availability>15500</availability>
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </delimiter>
        <timeZoneBias>480</timeZoneBias>
        <timeZoneName>Pacific Standard Time</timeZoneName>
        <device>deskphone</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </end>
      </state>
```

```
    </category>
    <category name="state" instance="1043521063" publishTime="2010-02-13T08:40:58.693"
container="2" version="30" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false" xsi:type="machineState">
        <availability>3500</availability>
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </delimiter>
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </end>
      </state>
    </category>
    <category name="state" instance="603979776" publishTime="2010-02-13T09:01:58.740"
container="2" version="1" expireType="time" expires="86402">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>9000</availability>
      </state>
    </category>
    <category name="dndState" instance="0" publishTime="2010-02-13T09:01:58.740"
container="2" version="6" expireType="static">
      <state xsi:type="userState" manual="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="3"
version="559" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" />
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
      </state>
    </category>
    <category name="state" instance="816379399" publishTime="2010-02-13T00:04:03.067"
container="3" version="102" expireType="endpoint" endpointId="5D4FEF3F-71C1-5562-A3D9-
56746B756BCB">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false" xsi:type="machineState">
        <availability>15500</availability>
        <delimiter xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/state"
xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </delimiter>
        <timeZoneBias>480</timeZoneBias>
        <timeZoneName>Pacific Standard Time</timeZoneName>
        <device>deskphone</device>
        <end xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/state"
xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </end>
      </state>
    </category>
    <category name="state" instance="1043521063" publishTime="2010-02-13T08:40:58.693"
container="3" version="30" expireType="endpoint" endpointId="B7D9FC43-19D7-5311-9E00-
C6C5B1D81BFB">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="false" xsi:type="machineState">
        <availability>3500</availability>
        <delimiter xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/state"
xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </delimiter>
        <device>computer</device>
        <end xmlns:auto-ns1="http://schemas.microsoft.com/2006/09/sip/state"
xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes">
        </end>
```

```
      </state>
    </category>
    <category name="state" instance="603979776" publishTime="2010-02-13T09:01:58.740"
container="3" version="1" expireType="time" expires="86402">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" manual="true" xsi:type="userState">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" minAvailability="6900"
maxAvailability="8999">
        </activity>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="100"
version="559" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9000</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2010-02-13T09:01:58.740"
container="100" version="559" expireType="user">
      <legacyInterop availability="9000" />
    </category>
    <category name="dndState" instance="0" publishTime="2010-02-13T09:01:58.740"
container="100" version="6" expireType="static">
      <state xsi:type="userState" manual="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="200"
version="559" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9000</availability>
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2010-02-13T09:01:58.740"
container="200" version="559" expireType="user">
      <legacyInterop availability="9000" />
    </category>
    <category name="dndState" instance="0" publishTime="2010-02-13T09:01:58.740"
container="200" version="6" expireType="static">
      <state xsi:type="userState" manual="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="300"
version="559" expireType="user">
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>6900</availability>
        <activity token="urgent-interruptions-only" />
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2010-02-13T09:01:58.740"
container="300" version="559" expireType="user">
      <legacyInterop availability="6900" token="urgent-interruptions-only" />
    </category>
    <category name="state" instance="1" publishTime="2010-02-13T09:01:58.740" container="400"
version="559" expireType="user">
```

```
      <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9000</availability>
        <delimiter xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
        <device>computer</device>
        <end xmlns="http://schemas.microsoft.com/2006/09/sip/commontypes" />
      </state>
    </category>
    <category name="legacyInterop" instance="1" publishTime="2010-02-13T09:01:58.740"
container="400" version="559" expireType="user">
      <legacyInterop availability="9000" />
    </category>
    <category name="dndState" instance="0" publishTime="2010-02-13T09:01:58.740"
container="400" version="6" expireType="static">
      <state xsi:type="userState" manual="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>9500</availability>
      </state>
    </category>
  </categories>
</roamingData>
```

4.4 Self SUBSCRIBE

A self-subscription dialog involves several different SIP messages. A user sends an initial self SUBSCRIBE request illustrated in section 4.4.1. The server sends a 200 OK response illustrated in section 4.4.2. As a result of publication, the server sends a BENOTIFY message to the endpoint registered to the self-subscribing user as illustrated in section 4.4.3. Finally, the self-subscribing user can request to cancel a self-subscription as illustrated in section 4.4.4.

4.4.1 Self SUBSCRIBE Request

A user sends out the initial self SUBSCRIBE request.

```
SUBSCRIBE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.32.208:2140
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=1415cd9c08;epid=ffad8b59dc
To: <sip:alice@contoso.com>
Call-ID: 0599fe5e383644229740f1afefa5b456
CSeq: 1 SUBSCRIBE
Contact: <sip:alice@contoso.com;opaque=user:epid:qKP5qGHu11azBsZ7CPso2AAA;gruu>
User-Agent: UCCP/2.0.6362.36 OC/2.0.6362.36 (Microsoft Office Communicator)
Event: vnd-microsoft-roaming-self
Accept: application/vnd-microsoft-roaming-self+xml
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: …
Content-Type: application/vnd-microsoft-roaming-self+xml
Content-Length: …

<roamingList xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self">
  <roaming type="categories"/>
  <roaming type="containers"/>
  <roaming type="subscribers"/>
  <roamingEx xmlns="http://schemas.microsoft.com/2007/09/sip/roaming-self-ex"
type="delegates"/>
</roamingList>
```

4.4.2 Self SUBSCRIBE 200 OK Response

The server then sends a 200 OK response, as follows:

```
SIP/2.0 200 OK
Contact:...
Authentication-Info: ...
Content-Length: …
From: "Alice"<sip:alice@contoso.com>;tag=1415cd9c08;epid=ffad8b59dc
To: <sip:alice@contoso.com>;tag=F63B0080
Call-ID: 0599fe5e383644229740f1afefa5b456
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 172.24.32.208:2140;ms-received-port=2140;ms-received-cid=5E00
Expires: 47519
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=47519
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
      xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
      xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
      xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
      xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:alice@contoso.com">
    <category name="calendarData" instance="0" publishTime="2008-01-11T17:10:45.107"
container="32000" version="1" expireType="static"/>
    <category name="calendarData" instance="0" publishTime="2008-01-11T17:10:45.107"
container="100" version="1" expireType="static"/>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="32000" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="400" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="300" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="200" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
```

```
        </identity>
      </contactCard>
    </category>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="100" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:07.890"
container="0" version="1" expireType="static">
      <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
        <identity>
          <name>
            <displayName>Alice</displayName>
          </name>
        </identity>
      </contactCard>
    </category>
    <category name="note" instance="0" publishTime="2008-01-11T17:10:45.107"
container="32000" version="1" expireType="static"/>
    <category name="note" instance="0" publishTime="2008-01-11T17:10:45.107"
container="100" version="1" expireType="static"/>
    <category name="state" instance="0" publishTime="2008-01-11T17:10:45.107"
container="32000" version="1" expireType="static">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             manual="false" xsi:type="aggregateState">
        <availability>18500</availability>
          <endpointLocation></endpointLocation>
      </state>
   </category>
    <category name="state" instance="0"
              publishTime="2008-01-22T23:40:49.543"
              container="400" version="1" expireType="static">
      <state xsi:type="aggregateState" lastActive="2008-01-22T23:40:49"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
      </state>
    </category>
    <category name="state" instance="0" publishTime="2008-01-22T23:40:49.543"
container="300" version="1" expireType="static">
      <state xsi:type="aggregateState" lastActive="2008-01-22T23:40:49"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
      </state>
    </category>
    <category name="state" instance="0"
              publishTime="2008-01-22T23:40:49.543"
              container="200" version="1" expireType="static">
      <state xsi:type="aggregateState"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
      </state>
    </category>
    <category name="state" instance="0"
              publishTime="2008-01-22T23:40:49.543"
              container="100" version="1" expireType="static">
      <state xsi:type="aggregateState"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
```

```
      </state>
    </category>
    <category name="state" instance="0"
              publishTime="2008-01-22T23:40:49.543"
              container="3" version="1" expireType="static">
      <state xsi:type="aggregateState" lastActive="2008-01-22T23:40:49"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
      </state>
    </category>
    <category name="state" instance="0"
              publishTime="2008-01-22T23:40:49.543"
              container="2" version="1" expireType="static">
      <state xsi:type="aggregateState" lastActive="2008-01-22T23:40:49"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>18000</availability>
      </state>
    </category>
    <category name="routing" instance="0"
              publishTime="2008-01-11T17:10:41.280" container="32000"
              version="1" expireType="static">
      <routing xmlns="http://schemas.microsoft.com/02/2006/sip/routing" name="rtcdefault"
version="1">
        <preamble>
          <flags name="protocol clientflags" value="block">
          </flags>
        </preamble>
      </routing>
    </category>
    <category name="legacyInterop" instance="0"
              publishTime="2008-01-11T17:06:07.890" container="32000"
              version="1" expireType="static">
      <legacyInterop availability="18500"/>
    </category>
    <category name="legacyInterop" instance="0"
              publishTime="2008-01-22T23:40:49.543" container="400"
              version="1" expireType="static">
      <legacyInterop availability="18000"/>
    </category>
    <category name="legacyInterop" instance="0"
              publishTime="2008-01-22T23:40:49.543" container="300"
              version="1" expireType="static">
      <legacyInterop availability="18000"/>
    </category>
    <category name="legacyInterop" instance="0"
              publishTime="2008-01-22T23:40:49.543" container="200"
              version="1" expireType="static">
      <legacyInterop availability="18000"/>
    </category>
    <category name="legacyInterop" instance="0"
              publishTime="2008-01-22T23:40:49.543" container="100"
              version="1" expireType="static">
      <legacyInterop availability="18000"/>
    </category>
    <category name="services" instance="0"
              publishTime="2008-01-11T17:10:45.107" container="32000"
              version="1" expireType="static">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      </services>
    </category>
    <category name="userProperties" instance="0"
              publishTime="2008-01-11T17:06:07.890" container="1"
              version="1" expireType="static">
      <userProperties>
        <telephonyMode>None</telephonyMode>
      </userProperties>
   </category>
```

```
   <category name="linkedPICContacts" instance="0" publishTime="2015-01-27T21:39:25.927"
container="1" version="57" expireType="static">
      <LinkedPicContactList>
        <LinkedPicContactEntry primary-Identity="bob@hotmail.com">
          <secondary-identities>
            <secondary-identity>bob@skypeids.net</secondary-identity>
            <secondary-identity>8d9a9d943984476c@skypecid.net</secondary-identity>
          </secondary-identities>
        </LinkedPicContactEntry>
        <LinkedPicContactEntry primary-Identity="alice@hotmail.com">
          <secondary-identities>
            <secondary-identity>alice@skypeids.net</secondary-identity>
            <secondary-identity>445abd8439841238@skypecid.net</secondary-identity>
          </secondary-identities>
        </LinkedPicContactEntry>
      </LinkedPicContactList>
    </category>
  </categories>
  <containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
    <container id="32000" version="0"/>
    <container id="400" version="0"/>
    <container id="300" version="0"/>
    <container id="200" version="1">
      <member type="sameEnterprise"/>
    </container>
    <container id="100" version="1">
      <member type="federated"/>
    </container>
    <container id="3" version="0"/>
    <container id="2" version="0"/>
    <container id="1" version="0"/>
    <container id="0" version="0">
      <member type="everyone"/>
    </container>
  </containers>
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
    <subscriber user="bob@contoso.com" displayName="Bob" acknowledged="false"
type="sameEnterprise"/>
  </subscribers>
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="1"/>
</roamingData>
```

4.4.3 Self SUBSCRIBE BENOTIFY

The server then sends a BENOTIFY on a self SUBSCRIBE dialog as a result of a publication.

```
BENOTIFY sip:172.24.32.208:2140;transport=tls;ms-opaque=37dc925222;ms-received-cid=5E00
SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:5061;branch=z9hG4bK0AAF5059.4FCDDC51;branched=FALSE
Authentication-Info: ...
Max-Forwards: 70
To: <sip:alice@contoso.com>;tag=1415cd9c08;epid=ffad8b59dc
Content-Length: …
From: <sip:alice@contoso.com>;tag=F63B0080
Call-ID: 0599fe5e383644229740f1afefa5b456
CSeq: 2 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=47515

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
```

```
xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:alice@contoso.com">
    <category name="state" instance="1" publishTime="2008-01-22T23:41:05.450" container="2"
version="1" expireType="user">
      <state xsi:type="aggregateState"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="state" instance="268435456"
              publishTime="2008-01-22T23:41:05.450" container="2"
              version="1" expireType="user">
      <state xsi:type="aggregateMachineState"
             endpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="state" instance="831107580"
              publishTime="2008-01-22T23:41:05.450" container="2"
              version="1" expireType="endpoint"
              endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
      <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             manual="false" xsi:type="machineState">
        <availability>3500</availability>
        <endpointLocation></endpointLocation>
      </state>
    </category>
    <category name="device" instance="412819398"
              publishTime="2008-01-22T23:41:05.450" container="2"
              version="1" expireType="endpoint"
              endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
      <device xmlns="http://schemas.microsoft.com/2006/09/sip/device"
              endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
        <capabilities preferred="false" uri="sip:alice@contoso.com">
          <text capture="true" render="true" publish="false">
          </text>
          <gifInk capture="false" render="true" publish="false">
          </gifInk>
          <isfInk capture="false" render="true" publish="false">
          </isfInk>
        </capabilities>
        <timezone>00:00:00-00:00</timezone>
        <machineName>MachineName</machineName>
      </device>
    </category>
    <category name="services" instance="0"
              publishTime="2008-01-22T23:41:05.450" container="2"
              version="1" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true"
                  publish="false"
                  preferredEndpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
                  deviceAvailability="3500"/>
            <gifInk render="true" capture="false"
                    publish="false"
                    preferredEndpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
                    deviceAvailability="3500"/>
            <isfInk render="true" capture="false"
                    publish="false"
                    preferredEndpointId="a8f9a3a8-ee61-56d7-b306-c67b08fb28d8"
                    deviceAvailability="3500"/>
          </capabilities>
```

```
    </service>
  </services>
</category>
<category name="state" instance="1"
          publishTime="2008-01-22T23:41:05.450" container="3"
          version="1" expireType="user">
  <state xsi:type="aggregateState"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>3500</availability>
  </state>
</category>
<category name="state" instance="831107580"
          publishTime="2008-01-22T23:41:05.450" container="3"
          version="1" expireType="endpoint"
          endpointId="A8F9A3A8-EE61-56D7-B306-C67B08FB28D8">
  <state xmlns="http://schemas.microsoft.com/2006/09/sip/state"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         manual="false" xsi:type="machineState">
    <availability>3500</availability>
    <endpointLocation></endpointLocation>
  </state>
</category>
<category name="state" instance="1"
          publishTime="2008-01-22T23:41:05.450" container="100"
          version="1" expireType="user">
  <state xsi:type="aggregateState"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>3500</availability>
  </state>
</category>
<category name="legacyInterop" instance="1"
          publishTime="2008-01-22T23:41:05.450" container="100"
          version="1" expireType="user">
  <legacyInterop availability="3500"/>
</category>
<category name="services" instance="0"
          publishTime="2008-01-22T23:41:05.450" container="100"
          version="1" expireType="user">
  <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
    <service uri="sip:alice@contoso.com">
      <capabilities>
        <text render="true" capture="true" deviceAvailability="3500"/>
        <gifInk render="true" capture="false" deviceAvailability="3500"/>
        <isfInk render="true" capture="false" deviceAvailability="3500"/>
      </capabilities>
    </service>
  </services>
</category>
<category name="state" instance="1"
          publishTime="2008-01-22T23:41:05.450" container="200"
          version="1" expireType="user">
  <state xsi:type="aggregateState"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://schemas.microsoft.com/2006/09/sip/state">
    <availability>3500</availability>
  </state>
</category>
<category name="legacyInterop" instance="1"
          publishTime="2008-01-22T23:41:05.450" container="200"
          version="1" expireType="user">
  <legacyInterop availability="3500"/>
</category>
<category name="services" instance="0"
          publishTime="2008-01-22T23:41:05.450" container="200"
          version="1" expireType="user">
  <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
    <service uri="sip:alice@contoso.com">
```

```
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500"/>
            <gifInk render="true" capture="false" deviceAvailability="3500"/>
            <isfInk render="true" capture="false" deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1"
              publishTime="2008-01-22T23:41:05.450" container="300"
              version="1" expireType="user">
      <state xsi:type="aggregateState"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1"
              publishTime="2008-01-22T23:41:05.450" container="300"
              version="1" expireType="user">
      <legacyInterop availability="3500"/>
    </category>
    <category name="services" instance="0"
              publishTime="2008-01-22T23:41:05.450" container="300"
              version="1" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500"/>
            <gifInk render="true" capture="false" deviceAvailability="3500"/>
            <isfInk render="true" capture="false" deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
    <category name="state" instance="1"
              publishTime="2008-01-22T23:41:05.450" container="400"
              version="1" expireType="user">
      <state xsi:type="aggregateState"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xmlns="http://schemas.microsoft.com/2006/09/sip/state">
        <availability>3500</availability>
      </state>
    </category>
    <category name="legacyInterop" instance="1"
              publishTime="2008-01-22T23:41:05.450" container="400"
              version="1" expireType="user">
      <legacyInterop availability="3500"/>
    </category>
    <category name="services" instance="0"
              publishTime="2008-01-22T23:41:05.450" container="400"
              version="1" expireType="user">
      <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
        <service uri="sip:alice@contoso.com">
          <capabilities>
            <text render="true" capture="true" deviceAvailability="3500"/>
            <gifInk render="true" capture="false" deviceAvailability="3500"/>
            <isfInk render="true" capture="false" deviceAvailability="3500"/>
          </capabilities>
        </service>
      </services>
    </category>
  </categories>
</roamingData>
```

The following example shows a BENOTIFY that is triggered by a **calendarState** publication operation:

```
BENOTIFY sip:172.24.32.124:54111;transport=tcp;ms-opaque=d445641ebd;ms-received-cid=1100
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK1A8DAF2D.8C27980A;branched=FALSE
Authentication-Info: ...
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=2e359b464c;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=1F700080
Call-ID: cccb942d2c044a8a8edb6828f7496d06
CSeq: 6 BENOTIFY
Require: eventlist
Content-Type: application/msrtc-event-categories+xml
Event: presence
subscription-state: active;expires=31927

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:user2@contoso.com">
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories" name="calendarData"
instance="2135971629" publishTime="2008-01-11T18:11:33.577">
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="user2@contoso.com">
      <freeBusy startTime="2008-01-10T08:00:00Z" granularity="PT15M" encodingVersion="1">
      </freeBusy>
    </calendarData>
  </category>
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories" name="calendarData"
instance="0" publishTime="2008-01-11T18:11:33.577">
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
mailboxID="user2@contoso.com">
      <WorkingHours xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
        <TimeZone>
          <Bias>480</Bias>
          <StandardTime>
            <Bias>0</Bias>
            <Time>02:00:00</Time>
            <DayOrder>1</DayOrder>
            <Month>11</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </StandardTime>
          <DaylightTime>
            <Bias>-60</Bias>
            <Time>02:00:00</Time>
            <DayOrder>2</DayOrder>
            <Month>3</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </DaylightTime>
        </TimeZone>
        <WorkingPeriodArray>
          <WorkingPeriod>
            <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
            <StartTimeInMinutes>600</StartTimeInMinutes>
            <EndTimeInMinutes>1140</EndTimeInMinutes>
          </WorkingPeriod>
        </WorkingPeriodArray>
      </WorkingHours>
    </calendarData>
  </category>
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories" name="contactCard"
instance="0" publishTime="2008-01-11T17:06:08.390">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>user2</displayName>
        </name>
      </identity>
    </contactCard>
  </category>
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories" name="state"
instance="1" publishTime="2008-01-11T18:11:33.577">
```

```
    <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>6500</availability>
      <activity token="in-a-meeting"/>
      <endpointLocation>Home_Custom_EndPoint_Location</endpointLocation>
      <meetingSubject>Customer Meeting</meetingSubject>
      <meetingLocation>Conference Room - Building 7</meetingLocation>
    </state>
  </category>
  <category xmlns="http://schemas.microsoft.com/2006/09/sip/categories" name="services"
instance="0" publishTime="2008-01-11T18:04:53.780">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:user2@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500"/>
          <gifInk render="true" capture="false" deviceAvailability="3500"/>
          <isfInk render="true" capture="false" deviceAvailability="3500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>
```

The following example is a self subscription NOTIFY that is generated as a result of a **setSubscriber** request:

```
BENOTIFY sip:172.24.32.124:54111;transport=tcp;ms-opaque=d445641ebd;ms-received-cid=1100
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bKBD3D826C.CFD7B7AF;branched=FALSE
Authentication-Info: ...
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=7b011632fc;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=C84D0080
Call-ID: f5d1bb95be0944dba41ef493a63383e4
CSeq: 14 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=45785

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"/>
</roamingData>
```

The following example is a self subscription NOTIFY body that is generated if the server did not remove **acknowledged** entries, or if the subscription that triggered the NOTIFY was an MSRTC or PIDF subscription:

```
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
    <subscriber user="john@contoso.com" displayName="User" acknowledged="true"
type="sameEnterprise" />
  </subscribers>
</roamingData>
```

4.4.4 Self SUBSCRIBE Cancel

The user cancels an earlier SUBSCRIBE request, as follows:

```
SUBSCRIBE sip:server.contoso.com;transport=tcp SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 2 SUBSCRIBE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: vnd-microsoft-roaming-self
Expires: 0
Accept: application/vnd-microsoft-roaming-self+xml
Proxy-Authorization: ...
Content-Length: 0
```

The server sends a 200 OK response, as follows:

```
SIP/2.0 200 OK
Contact: <sip:server.contoso.com;transport=tcp>
Authentication-Info: ...
From: "Bob"<sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 2 SUBSCRIBE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms-received-cid=B00
Expires: 0


 Content-Length: 0
```

4.5 Batched Category SUBSCRIBE

The user sends an initial batched category SUBSCRIBE request.

```
SUBSCRIBE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:1596
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 1 SUBSCRIBE
Contact: <sip:alice@contoso.com;opaque=user:epid:YoF3PgQjXFaEdxpfElfutwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml,
text/xml+msrtc.pidf, application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: ...
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:alice@contoso.com" name="">
  <action name="subscribe" id="1177624">
    <adhocList>
      <resource uri="sip:john@contoso.com"/>
```

```
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

The server sends back a 200 OK response.

```
SIP/2.0 200 OK
Contact: …
Authentication-Info: …
Content-Length: …
From: "Alice"<sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=C9230080
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 172.24.41.100:1596;ms-received-port=1596;ms-received-cid=6B00
Expires: 34560
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=d20e87acad9e400aac184344edb29a4a
Event: presence
subscription-state: active;expires=34560
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

--d20e87acad9e400aac184344edb29a4a
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:alice@contoso.com" version="0"
fullState="false"/>

--d20e87acad9e400aac184344edb29a4a
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:john@contoso.com">
  <category name="calendarData" instance="0" publishTime="2008-01-11T18:28:58.030">
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
                  mailboxID="john@contoso.com">
      <WorkingHours xmlns="http://schemas.microsoft.com/exchange/services/2006/types">
        <TimeZone>
          <Bias>480</Bias>
          <StandardTime>
            <Bias>0</Bias>
            <Time>02:00:00</Time>
            <DayOrder>1</DayOrder>
            <Month>11</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </StandardTime>
          <DaylightTime>
            <Bias>-60</Bias>
            <Time>02:00:00</Time>
            <DayOrder>2</DayOrder>
            <Month>3</Month>
            <DayOfWeek>Sunday</DayOfWeek>
          </DaylightTime>
        </TimeZone>
        <WorkingPeriodArray>
```

```
          <WorkingPeriod>
            <DayOfWeek>Monday Tuesday Wednesday Thursday Friday</DayOfWeek>
            <StartTimeInMinutes>600</StartTimeInMinutes>
            <EndTimeInMinutes>1140</EndTimeInMinutes>
          </WorkingPeriod>
        </WorkingPeriodArray>
      </WorkingHours>
    </calendarData>
  </category>
  <category name="contactCard" instance="0" publishTime="2008-01-22T23:41:04.577">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>john</displayName>
        </name>
      </identity>
    </contactCard>
  </category>
  <category name="note"/>
  <category name="state" instance="1" publishTime="2008-01-24T02:04:35.327">
    <state xsi:type="aggregateState"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>6500</availability>
    </state>
  </category>
  <category name="services" instance="0"
            publishTime="2008-01-24T02:04:35.327">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:john@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500"/>
          <gifInk render="true" capture="false" deviceAvailability="3500"/>
          <isfInk render="true" capture="false" deviceAvailability="3500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>

--d20e87acad9e400aac184344edb29a4a--
```

The user then adds another user to the existing batch subscription dialog.

```
SUBSCRIBE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:1596
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=C9230080
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 2 SUBSCRIBE
Contact: <sip:alice@contoso.com;opaque=user:epid:YoF3PgQjXFaEdxpfElfutwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml,
text/xml+msrtc.pidf, application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: …
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:alice@contoso.com" name="">
```

```
  <action name="subscribe" id="1177624">
    <adhocList>
      <resource uri="sip:bob@contoso.com">
        <context/>
      </resource>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

The server (2) sends back a 200 OK response.

```
SIP/2.0 200 OK
Contact: <sip:server.contoso.com:5061;transport=tls>
Authentication-Info: …
Content-Length: …
From: "Alice"<sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=C9230080
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 2 SUBSCRIBE
Via: SIP/2.0/TLS 172.24.41.100:1596;ms-received-port=1596;ms-received-cid=6B00
Expires: 22752
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=b6778ced37dc4bb1b40bbfd006a6beae
Event: presence
subscription-state: active;expires=22752
ms-piggyback-cseq: 2
Supported: ms-benotify, ms-piggyback-first-notify

--b6778ced37dc4bb1b40bbfd006a6beae
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:alice@contoso.com" version="0"
fullState="false"/>

--b6778ced37dc4bb1b40bbfd006a6beae
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
  <category name="calendarData" instance="2135971629" publishTime="2008-01-
19T17:58:08.623">
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
                  mailboxID="bob@contoso.com">
      <freeBusy startTime="2008-01-18T08:00:00Z" granularity="PT15M" encodingVersion="1">
      </freeBusy>
    </calendarData>
  </category>
  <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:08.077">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>Bob</displayName>
        </name>
      </identity>
```

```
    </contactCard>
  </category>
  <category name="contactCard" instance="1" publishTime="2008-01-11T17:28:18.530">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <phone type="work">
        <uri>tel:5558828089;phone-context=dialstring</uri>
      </phone>
      <phone type="mobile">
        <uri>tel:5558828081;phone-context=dialstring</uri>
      </phone>
    </contactCard>
  </category>
  <category name="note"/>
  <category name="state" instance="1" publishTime="2008-01-22T19:58:38.467">
    <state xsi:type="aggregateState" lastActive="2008-01-22T19:58:38"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>15500</availability>
    </state>
  </category>
  <category name="services" instance="0" publishTime="2008-01-22T19:58:38.467">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:bob@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="15500"/>
          <gifInk render="true" capture="false" deviceAvailability="15500"/>
          <isfInk render="true" capture="false" deviceAvailability="15500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>
```

The user removes all category subscriptions for john@contoso.com from the dialog.

```
SUBSCRIBE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:1596
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=C9230080
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 3 SUBSCRIBE
Contact: <sip:alice@contoso.com;opaque=user:epid:YoF3PgQjXFaEdxpfElfutwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml,
text/xml+msrtc.pidf, application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: …
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:alice@contoso.com" name="">
  <action name="unsubscribe" id="1177624">
    <adhocList>
      <resource uri="sip:john@contoso.com"/>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
```

```
        <category name="state"/>
      </categoryList>
    </action>
  </batchSub>
```

The server sends back a 200 OK response.

```
SIP/2.0 200 OK
Contact: <sip:server.contoso.com:5061;transport=tls>
Authentication-Info: …
Content-Length: …
From: "Alice"<sip:alice@contoso.com>;tag=09fedac5c9;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=C9230080
Call-ID: 1de296daa57a419c9006fdc51b33ed8b
CSeq: 3 SUBSCRIBE
Via: SIP/2.0/TLS 172.24.41.100:1596;ms-received-port=1596;ms-received-cid=6B00
Expires: 27936
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=626cda06b25c49bcabe5d62d771dbdf3
Event: presence
subscription-state: active;expires=27936
ms-piggyback-cseq: 3
Supported: ms-benotify, ms-piggyback-first-notify


--626cda06b25c49bcabe5d62d771dbdf3
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:alice@contoso.com" version="0"
fullState="false"/>

--626cda06b25c49bcabe5d62d771dbdf3--
```

4.6 Single Category SUBSCRIBE

Bob@contoso.com sends the initial batch subscribe to john@fabrikam.com, as follows:

```
SUBSCRIBE sip:bob@contoso.com SIP/2.0


Via: SIP/2.0/TLS 10.4.22.76:2168
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=0c33196f2d;epid=0b85fd1bd6
To: <sip:bob@contoso.com>
Call-ID: f43a56d4771544499ae40a39e5dc039d
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@contoso.com;opaque=user:epid:tDDhdAyfsFq_rGXvlxZSLQAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml, text/xml+msrtc.pidf,
application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: ...
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …
```

```
<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:bob@contoso.com" name="">
  <action name="subscribe" id="19254600">
    <adhocList>
      <resource uri="sip:john@fabrikam.com"/>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

The server returns a 200 OK message with **state** set to "resubscribe" in the response, as follows:

```
SIP/2.0 200 OK
Contact: <sip:B04-OCG.contoso.com:5061;transport=tls>
Authentication-Info: ...
Content-Length: …
From: "Bob"<sip:bob@contoso.com>;tag=0c33196f2d;epid=0b85fd1bd6
To: <sip:bob@contoso.com>;tag=DB120080
Call-ID: f43a56d4771544499ae40a39e5dc039d
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 10.4.22.76:2168;ms-received-port=2168;ms-received-cid=1000
Expires: 0
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=be3738dd822646dc848aef2b5dd10bf3
Event: presence
subscription-state: terminated;expires=0
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

--be3738dd822646dc848aef2b5dd10bf3
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:bob@contoso.com" version="0"
fullState="false">
  <resource uri="sip:john@fabrikam.com">
    <instance id="0" state="resubscribe" cid="john@fabrikam.com"/>
  </resource>
</list>

--be3738dd822646dc848aef2b5dd10bf3--
```

The user bob@contoso.com then sends a single SUBSCRIBE request to john@fabrikam.com, as follows:

```
SUBSCRIBE sip:john@fabrikam.com SIP/2.0
Via: SIP/2.0/TLS 10.4.22.76:2168
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=6f0237b4ea;epid=0b85fd1bd6
To: <sip:john@fabrikam.com>
Call-ID: 911a493935c449a0a104de04bb2911e7
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@contoso.com;opaque=user:epid:tDDhdAyfsFq_rGXvlxZSLQAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
```

```
Accept: application/msrtc-event-categories+xml, application/xpidf+xml, text/xml+msrtc.pidf,
application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: ...
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:bob@contoso.com" name="">
  <action name="subscribe" id="1150776">
    <adhocList>
      <resource uri="sip:john@fabrikam.com"/>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

The SUBSCRIBE request is routed to the fabrikam.com domain, and a 200 OK message is returned with john@fabrikam.com's categories, as follows:

```
SIP/2.0 200 OK
Authentication-Info: ...
ms-edge-proxy-message-trust: ms-source-type=DirectPartner;ms-ep-fqdn=b10-
ocg.contoso.com;ms-source-verified-user=verified;ms-source-network=federation
Contact: <sip:conf.fabrikam.com:5061;transport=tls;ms-fe=D15-OCG.fabrikam.com>
Content-Length: …
From: "bob"<sip:bob@contoso.com>;tag=6f0237b4ea;epid=0b85fd1bd6
To: <sip:john@fabrikam.com>;tag=49160080
Call-ID: 911a493935c449a0a104de04bb2911e7
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 10.4.22.76:2168;ms-received-port=2168;ms-received-cid=1000
Record-Route:...
Expires: 30816
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=ab448217ad354b6784e06462478d9a45
Event: presence
subscription-state: active;expires=30816
ms-piggyback-cseq: 1
Supported: ms-piggyback-first-notify
Record-Route: <sip:B04-OCG.contoso.com:5061;transport=tls;ms-role-rs-from;lr;ms-rgs-
cid=1000;ms-route-sig=baJBrBacIpyYMZlY55dZXW8cJ8rFLqKLNOFqVWGwAA>

--ab448217ad354b6784e06462478d9a45
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:bob@contoso.com" version="0"
fullState="false"/>

--ab448217ad354b6784e06462478d9a45
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:john@fabrikam.com">
  <category name="calendarData" instance="0" publishTime="2008-01-25T21:59:33.093"/>
  <category name="contactCard" instance="0" publishTime="2008-01-17T00:29:56.510">
```

```
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>John</displayName>
        </name>
        <email>john@fabrikam.com</email>
      </identity>
    </contactCard>
  </category>
  <category name="note" instance="0" publishTime="2008-01-25T21:59:33.093"/>
  <category name="state" instance="1" publishTime="2008-01-25T22:30:28.973">
    <state xsi:type="aggregateState" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>3500</availability>
    </state>
  </category>
  <category name="services" instance="0" publishTime="2008-01-25T22:29:59.753">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:john@fabrikam.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500"/>
          <gifInk render="true" capture="false" deviceAvailability="3500"/>
          <isfInk render="true" capture="false" deviceAvailability="3500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>

--ab448217ad354b6784e06462478d9a45--
```

4.7 Polling Category SUBSCRIBE

The following example shows a polling batch category SUBSCRIBE request. The combination of the **Expires** header value ("0") and the absence of the **tag** parameter in the **To** header indicates a polling subscribe, as a result of which the server does NOT create a dialog.

```
SUBSCRIBE sip:john@contoso.com SIP/2.0
Via: SIP/2.0/TCP 10.88.20.31:3807
Max-Forwards: 70
From: <sip:john@contoso.com>;tag=e92e3c8e01;epid=c05eb044ab
To: <sip:john@contoso.com>
Call-ID: 2cca5716b69d404cab405f2d10b45a7c
CSeq: 1 SUBSCRIBE
Contact: <sip:john@contoso.com;opaque=user:epid:fdYQyOcdZ1qt YAFbFcqowAA;gruu>
User-Agent: UCCP/2.0.6362.13 OC/2.0.6362.13 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml, text/xml+msrtc.pidf,
application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Expires: 0
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:john@contoso.com" name="">
  <action name="subscribe" id="95865280">
    <adhocList>
      <resource uri="sip:bob@contoso.com"/>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="services"/>
      <category name="state"/>
```

```
      </categoryList>
    </action>
  </batchSub>
```

The server sends back a 200 OK response, as follows:

```
SIP/2.0 200 OK
Contact: <sip:server.contoso.com;transport=tcp>
Authentication-Info: …
From: "john"<sip:john@contoso.com>;tag=e92e3c8e01;epid=c05eb044ab
To: <sip:john@contoso.com>;tag=3B320080
Call-ID: 2cca5716b69d404cab405f2d10b45a7c
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TCP 10.88.20.31:3807;ms-received-port=3807;ms-received-cid=1500
Expires: 0
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=d1253170c4b643f9b930a5a191748c1c
Event: presence
subscription-state: terminated;expires=0
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

--d1253170c4b643f9b930a5a191748c1c
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml
<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:john@contoso.com" version="0"
fullState="false"/>

--d1253170c4b643f9b930a5a191748c1c
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml
<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
  <category name="services" instance="0" publishTime="2008-01-11T18:25:58.543">
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:bob@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="3500"/>
          <gifInk render="true" capture="false" deviceAvailability="3500"/>
          <isfInk render="true" capture="false" deviceAvailability="3500"/>
        </capabilities>
      </service>
    </services>
  </category>
  <category name="state" instance="1" publishTime="2008-01-11T18:26:06.450">
    <state xsi:type="aggregateState"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>9000</availability>
    </state>
  </category>
</categories>

--d1253170c4b643f9b930a5a191748c1c--
```

4.8 setSubscriber

The user, alice@contoso.com, adds bob@contoso.com as a contact, as shown in the following diagram.

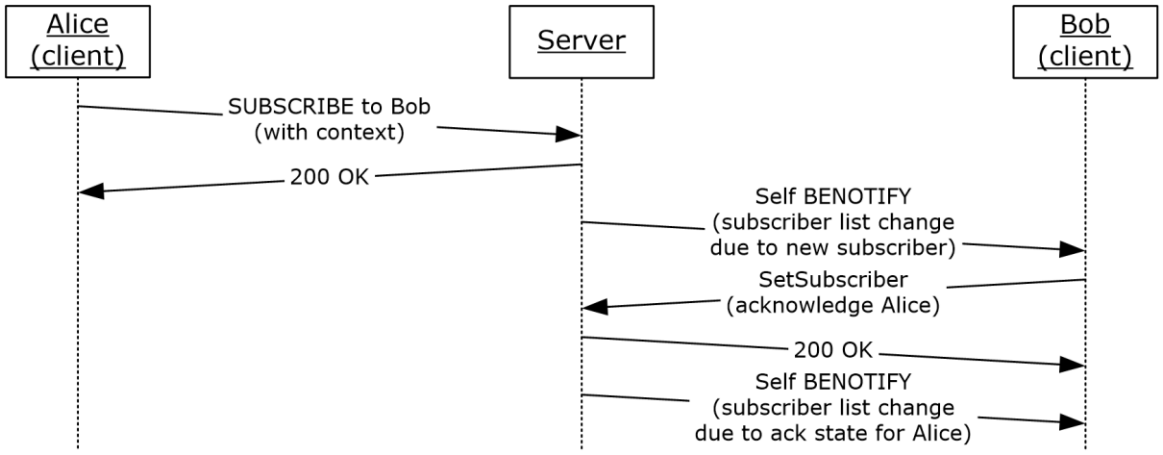

Figure 13: Sample call flow for setSubscriber

The SIP protocol client sends a SUBSCRIBE request using a **context** element so that the publisher is notified of the contact addition, as showing in the following example:

```
SUBSCRIBE sip:alice@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:1706
Max-Forwards: 70
From: <sip:alice@contoso.com>;tag=62ec6bdbd4;epid=5e434f57d3
To: <sip:alice@contoso.com>
Call-ID: f285622ca97b462e9db25aa95a71d12e
CSeq: 1 SUBSCRIBE
Contact: <sip:alice@contoso.com;opaque=user:epid:YoF3PgQjXFaEdxpfElfutwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: presence
Accept: application/msrtc-event-categories+xml, application/xpidf+xml, text/xml+msrtc.pidf,
application/pidf+xml, application/rlmi+xml, multipart/related
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Require: adhoclist, categoryList
Supported: eventlist
Proxy-Authorization: …
Content-Type: application/msrtc-adrl-categorylist+xml
Content-Length: …

<batchSub xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
uri="sip:alice@contoso.com" name="">
  <action name="subscribe" id="34111920">
    <adhocList>
      <resource uri="sip:bob@contoso.com">
        <context/>
      </resource>
    </adhocList>
    <categoryList xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist">
      <category name="calendarData"/>
      <category name="contactCard"/>
      <category name="note"/>
      <category name="services"/>
      <category name="state"/>
    </categoryList>
  </action>
</batchSub>
```

The server sends back a 200 OK response, as follows:

```
SIP/2.0 200 OK
Contact: <sip:server.contoso.com:5061;transport=tls>
Authentication-Info: …
Content-Length: …
From: "Alice"<sip:alice@contoso.com>;tag=62ec6bdbd4;epid=5e434f57d3
To: <sip:alice@contoso.com>;tag=3B2C0080
Call-ID: f285622ca97b462e9db25aa95a71d12e
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 172.24.41.100:1706;ms-received-port=1706;ms-received-cid=6D00
Expires: 25344
Require: eventlist
Content-Type: multipart/related; type="application/rlmi+xml";start=resourceList;
boundary=77ab14f3e1e84a049dcec5d8358b2ff9
Event: presence
subscription-state: active;expires=25344
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify


--77ab14f3e1e84a049dcec5d8358b2ff9
Content-Transfer-Encoding: binary
Content-ID: resourceList
Content-Type: application/rlmi+xml

<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:alice@contoso.com" version="0"
fullState="false"/>

--77ab14f3e1e84a049dcec5d8358b2ff9
Content-Transfer-Encoding: binary
Content-Type: application/msrtc-event-categories+xml

<categories xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
uri="sip:bob@contoso.com">
  <category name="calendarData" instance="2135971629" publishTime="2008-01-
19T17:58:08.623">
    <calendarData xmlns="http://schemas.microsoft.com/2006/09/sip/calendarData"
                  mailboxID="bob@contoso.com">
      <freeBusy startTime="2008-01-18T08:00:00Z" granularity="PT15M" encodingVersion="1">
      </freeBusy>
    </calendarData>
  </category>
  <category name="contactCard" instance="0" publishTime="2008-01-11T17:06:08.077">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <identity>
        <name>
          <displayName>Bob</displayName>
        </name>
      </identity>
    </contactCard>
  </category>
  <category name="contactCard" instance="1" publishTime="2008-01-11T17:28:18.530">
    <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
      <phone type="work">
        <uri>tel:5558828089;phone-context=dialstring</uri>
      </phone>
      <phone type="mobile">
        <uri>tel:5558828081;phone-context=dialstring</uri>
      </phone>
    </contactCard>
  </category>
  <category name="note"/>
  <category name="state" instance="1" publishTime="2008-01-22T19:58:38.467">
    <state xsi:type="aggregateState" lastActive="2008-01-22T19:58:38"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns="http://schemas.microsoft.com/2006/09/sip/state">
      <availability>15500</availability>
    </state>
  </category>
  <category name="services" instance="0" publishTime="2008-01-22T19:58:38.467">
```

```
    <services xmlns="http://schemas.microsoft.com/2006/09/sip/service">
      <service uri="sip:bob@contoso.com">
        <capabilities>
          <text render="true" capture="true" deviceAvailability="15500"/>
          <gifInk render="true" capture="false" deviceAvailability="15500"/>
          <isfInk render="true" capture="false" deviceAvailability="15500"/>
        </capabilities>
      </service>
    </services>
  </category>
</categories>
```

The following example shows a self SUBSCRIBE NOTIFY request for bob@contoso.com.

The preceding SUBSCRIBE request with a **context** element causes a subscriber entry for alice@contoso.com to be created in Bob's subscriber list. This action triggers a self SUBSCRIBE NOTIFY with "subscribers" scope, as specified in section 3.4.5.1.

```
BENOTIFY sip:172.24.41.100:1710;transport=tls;ms-opaque=c455c01228;ms-received-cid=6F00
SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:5061;branch=z9hG4bKDA1327AE.8699748D;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=676f7038e7;epid=5e434f57d3
Content-Length: …
From: <sip:bob@contoso.com>;tag=BF0F0080
Call-ID: b3840f60a46542d7ab597d8c3466310a
CSeq: 3 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=48355

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
    <subscriber user="alice@contoso.com" displayName="Alice" acknowledged="false"
type="sameEnterprise"/>
  </subscribers>
</roamingData>
```

4.8.1 A setSubscribers Request

After bob@contoso.com receives the self SUBSCRIBE NOTIFY, he sends a **setSubscribers** request to acknowledge alice@contoso.com.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:1710
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=3c8e860cc0;epid=5e434f57d3
To: <sip:bob@contoso.com>
Call-ID: dddea4396b6049078c9624f01b73f12b
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:YoF3PgQjXFaEdxpfElfutwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-presence-setsubscriber+xml
Content-Length: …

<setSubscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscriber user="alice@contoso.com" acknowledged="true"/>
```

```
</setSubscribers>
```

The server sends back a 200 OK response.

```
SIP/2.0 200 OK
Authentication-Info: …
From: "Bob"<sip:bob@contoso.com>;tag=3c8e860cc0;epid=5e434f57d3
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: dddea4396b6049078c9624f01b73f12b
CSeq: 1 SERVICE
Via: SIP/2.0/TLS 172.24.41.100:1710;ms-received-port=1710;ms-received-cid=6F00
Content-Length: 0
```

A Self SUBSCRIBE NOTIFY request results from a change in "subscribers" scope because of a **setSubscribers** operation.

Any change in subscriber list, which in this case is because of a **setSubscribers** request, causes a self SUBSCRIBE NOTIFY request to be generated.

```
BENOTIFY sip:172.24.41.100:1710;transport=tls;ms-opaque=c455c01228;ms-received-cid=6F00
SIP/2.0
Via: SIP/2.0/TLS 172.24.41.100:5061;branch=z9hG4bKDA1327AE.8699748D;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=676f7038e7;epid=5e434f57d3
Content-Length: …
From: <sip:bob@contoso.com>;tag=BF0F0080
Call-ID: b3840f60a46542d7ab597d8c3466310a
CSeq: 3 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=48355

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <subscribers xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"/>
</roamingData>
```

4.9 setContainerMembers

4.9.1 setContainerMembers Request

The following example demonstrates a **setContainerMembers** request by bob@contoso.com to add member john@contoso.com to container 32000:

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=bb0b6e6dc7;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 7dd9a94511204aa6a742d8951a765a1f
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt -AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-setcontainermembers+xml
Content-Length: …
```

```
<setContainerMembers xmlns="http://schemas.microsoft.com/2006/09/sip/container-management">
  <container id="32000" version="0">
    <member action="add" type="user" value="john@contoso.com" />
  </container>
</setContainerMembers>
```

The server sends a 200 OK response to bob@contoso.com.

```
SIP/2.0 200 OK
Authentication-Info: …
From: "Bob"<sip:bob@contoso.com>;tag=bb0b6e6dc7;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 7dd9a94511204aa6a742d8951a765a1f
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms-received-cid=B00
Content-Length: 0
```

The server sends the resulting BENOTIFY on active self-subscription dialog boxes, as follows:

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK46F0BC36.0E24F0FE;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 16 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46449
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
    <container id="32000" version="1">
      <member type="user" value="john@contoso.com" />
    </container>
  </containers>
</roamingData>
```

4.9.2 Removing a Container Member

This example demonstrates a **setContainerMembers** request by bob@contoso.com to remove member john@contoso.com from container 32000.

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>;tag=4d9279d056;epid=84d3db8c23
To: <sip:bob@contoso.com>
Call-ID: 409fed08bd7a4f83a64ec43404476201
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-setcontainermembers+xml
Content-Length: …
<setContainerMembers xmlns="http://schemas.microsoft.com/2006/09/sip/container-management">
```

```
  <container id="32000" version="1">
    <member action="remove" type="user" value="john@contoso.com" />
  </container>
</setContainerMembers>
```

The server sends a 200 OK response to bob@contoso.com, as follows:

```
SIP/2.0 200 OK
Authentication-Info: …
From: "Bob"<sip:bob@contoso.com>;tag=4d9279d056;epid=84d3db8c23
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: 409fed08bd7a4f83a64ec43404476201
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms-received-cid=B00
Content-Length: 0
```

The 200 OK response results in a BENOTIFY from the server on active self subscription dialogs.

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK47F0BC36.767A2B06;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 17 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46434
<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
           xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
           xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
           xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers">
  <containers xmlns="http://schemas.microsoft.com/2006/09/sip/containers">
    <container id="32000" version="2" />
  </containers>
</roamingData>
```

4.10 PIDF/RPID SUBSCRIBE

A federated user, alice@fabrikam.com, subscribes to presence for bob@contoso.com according to the semantics of [RFC3265] with **Event**: presence and **Accept** type as "application/pidf+xml".

```
SUBSCRIBE sip:bob@contoso.com;maddr=server.contoso.com SIP/2.0
From: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
To: <sip:bob@contoso.com>
CSeq: 1 SUBSCRIBE
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Max-Forwards: 67
Contact: <sip:server.fabrikam.com:5061;maddr=server.fabrikam.com;transport=TLS>
Expires: 16406
Accept: application/pidf+xml
Event: presence
Content-Length: 0
```

The server sends a 200 OK acknowledging the receipt of the subscription, as follows:

```
SIP/2.0 200 OK
From: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
To: <sip:bob@contoso.com>;tag=6667115B
CSeq: 1 SUBSCRIBE
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Expires: 16406
Content-Length: 0
```

The server sends a NOTIFY request with the presence of the user bob@contoso.com, as follows:

```
NOTIFY sip:server.fabrikam.com:5061;maddr=server.fabrikam.com;transport=TLS SIP/2.0
From: <sip:bob@contoso.com>;tag=6667115B
To: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
CSeq: 1 NOTIFY
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Max-Forwards: 70
Content-Length: …
Content-Type: application/pidf+xml
Event: presence
subscription-state: active;expires=16406
…
<?xml version="1.0" encoding="utf-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
          xmlns:ep="urn:ietf:params:xml:ns:pidf:status:rpid-status"
          xmlns:et="urn:ietf:params:xml:ns:pidf:rpid-tuple"
          xmlns:ci="urn:ietf:params:xml:ns:pidf:cipid"
          entity="sip:bob@contoso.com">
  <tuple id="0">
    <status>
      <basic>open</basic>
      <ep:activities>
        <ep:activity>away</ep:activity>
      </ep:activities>
    </status>
  </tuple>
  <linked-identities primary-id="bob@ contoso.com">
    <secondary-identities>
      <secondary-identity>bob@skypeids.net</secondary-identity>
      <secondary-identity>A1519ABC8FAC5FA71@skypecid.net</secondary-identity>
    </secondary-identities>
  </linked-identities>
  <cid>A1519ABC8FAC5FA71</cid>
  <ci:display-name>Bob at Contoso</ci:display-name>
</presence>
```

The subscribing SIP protocol client acknowledges the receipt of the notification with a 200 OK, as follows:

```
SIP/2.0 200 OK
From: <sip:bob@contoso.com>;tag=6667115B
To: <sip:alice@fabrikam.com>;tag=0-13c4-4796af3d-1f212e63-62fcb2b9
CSeq: 1 NOTIFY
Call-ID: 60ae14f0-0-13c4-4796af3d-1f212e62-6a76c503@fabrikam.com
Content-Length: 0
```

4.11 MSRTC SUBSCRIBE

A federated user, alice@fabrikam.com, subscribes to the presence of bob@contoso.com, as follows:

```
SUBSCRIBE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TLS 10.46.164.196:1620
Max-Forwards: 70
```

```
From: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
To: <sip:bob@contoso.com>
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 1 SUBSCRIBE
Contact: <sip:alice@fabrikam.com:1620;maddr=10.46.164.196;transport=tls>;proxy=replace
User-Agent: LCC/1.3
Event: presence
Accept: application/xpidf+xml, text/xml+msrtc.pidf, application/pidf+xml
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: ...
Content-Length: 0
```

The server sends a **presence** document in a 200 OK response, as follows:

```
SIP/2.0 200 OK
Authentication-Info: ...
ms-edge-proxy-message-trust: ms-source-type=AutoFederation;ms-ep-fqdn=lcsproxy-
internal.fabrikam.com;ms-source-verified-user=verified;ms-source-network=federation
Contact: <sip:ocsserver.contoso.com:5061;transport=tls>
Content-Length: …
From: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
To: <sip:bob@contoso.com>;tag=B8FCF750
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 10.46.164.196:1620;ms-received-port=1620;ms-received-cid=1dcee00
Record-Route: …
Expires: 29087
Content-Type: text/xml+msrtc.pidf
Event: presence
subscription-state: active;expires=29087
ms-piggyback-cseq: 1
Supported: ms-piggyback-first-notify
Record-Route: <sip:sipfed.contoso.com:5061;transport=tls;lr;ms-key-info=jACAAPo9w-
9gLtcJgl_IAQECAAADZgAAAKQAAAnTaukuFFMFe_Sxt9an2NqgIu2oegkAvaJbqlQeT1caGA42DMKNbTe7VTqOXAIod
kufX0rROiiPCuu5KfsMgjduDjdVTqO-gB 0RrVrnz0KlCg c2iFYhp82puKA7aiQG32qXWbL6DRGe-
05AY4672ZKfftcjOvngToyO4VY3NClKyUGN0DKQZ89Ac0ime6wNSgJuZloJzYxdJ-2hOddU_-
nS3xW_1xbpz2XC1jcAlJTzNml8HWp6zi9KOOoi18TwBGADRMLXrMnUFCaQnCbVf49fzNTKoM5rCihcshCk1GYVH3ClR
bzjwgmx3lLv6HrQ5mrBm5mqwM2tV1a9ednogA;ms-route-
sig=bkBvoYzffbAVQtNIHC7e q2qt z4hWPI3VZvzOVwAA>
Record-Route: <sip:lcsproxy-internal.fabrikam.com:5061;transport=tls;lr>
Record-Route: <sip:NALCSDIRECT01.na.fabrikam.com:5061;transport=tls;lr>
Record-Route: <sip:nalcspool.na.fabrikam.com:5061;transport=tls;ms-
fe=NALCSFE02.na.fabrikam.com;lr;ms-route-sig=cadNbjDLZFNRAuNoAxZtyRASyXIfE->
<presentity uri="bob@contoso.com" xmlns=
            "http://schemas.microsoft.com/2002/09/sip/presence"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <availability aggregate="300" />
  <activity aggregate="400" />
  <displayName displayName="Bob" />
  <email email="bob@contoso.com" />
  <phoneNumber number="+1 (123) 4567890 X67890" />
  <aggregate>
    <states>
      <state avail="3500" xsi:type="userState">online</state>
    </states>
  </aggregate>
</presentity>
```

The server sends a NOTIFY request when the presence of bob@contoso.com changes, as shown in the following example:

```
NOTIFY sip:10.46.164.196:1620;transport=tls;ms-received-cid=1DCEE00 SIP/2.0
```

```
Authentication-Info: …
Via: SIP/2.0/TLS 172.30.36.108:5061;branch=z9hG4bK622E3399.79971B7B;branched=FALSE;ms-
internal-info="aan8w2KNqmlXBytq9eqdN9b-TX7pkA"
Max-Forwards: 64
Via: …
To: <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
Content-Length: …
From: <sip:bob@contoso.com>;tag=B8FCF750
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 2 NOTIFY
Content-Type: text/xml+msrtc.pidf
Event: presence
subscription-state: active;expires=27503
<presentity uri="bob@contoso.com"
            xmlns="http://schemas.microsoft.com/2002/09/sip/presence"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <availability aggregate="300" />
  <activity aggregate="600" />
  <displayName displayName="Bob" />
  <email email="bob@contoso.com" />
  <phoneNumber number="+1 (123) 4567890 X67890" />
  <aggregate>
    <states>
      <state avail="6500" xsi:type="userState">busy</state>
    </states>
  </aggregate>
</presentity>
```

The SIP protocol client sends a 200 OK response for the NOTIFY request, as follows:

```
SIP/2.0 200 OK
Via: …
From: <sip:bob@contoso.com>;tag=B8FCF750
To: "Alice" <sip:alice@fabrikam.com>;tag=38209ebf3e;epid=d6277c8671
Call-ID: 23ffb7f278764277871ff5b4ba90ae11
CSeq: 2 NOTIFY
User-Agent: LCC/1.3
Proxy-Authorization: ...
Content-Length: 0
```

4.12 setDelegate

4.12.1 Adding a New Delegate

The following example shows user bob@contoso.com adding a new delegate john@contoso.com to his list of delegates:

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
From: <sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu="">
User-Agent: UCCP/3.0.6789.0 OC/3.0.6789.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-setDelegate +xml
Content-Length: …

<setDelegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegate-management"
version="3">
  <delegate uri="sip:john@contoso.com" action="add"/>
```

```
</setDelegates>
```

The server sends a 200 OK response to bob@contoso.com, as follows:

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms
received-cid=B00
Content-Type: application/vnd-microsoft-roaming-self+xml

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
 xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
 xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
 xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
 xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="4">
    <delegate uri="john@contoso.com" publish="false" redelegate="false"/>
  </delegates>
</roamingData>
```

The server then sends out a notification on all active self subscription dialogs, as shown in the following example:

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK46F0BC36.0E24F0FE;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 16 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46449

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
 xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
 xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
 xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
 xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="4">
    <delegate uri="john@contoso.com" publish="false" redelegate="false"/>
  </delegates>
</roamingData>
```

4.12.2 Removing an Existing Delegate

The following example shows how bob@contoso.com will remove john@contoso.com from the list of delegates:

```
SERVICE sip:bob@contoso.com SIP/2.0
Via: SIP/2.0/TCP 172.24.32.124:53925
Max-Forwards: 70
```

```
From: <sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Contact: <sip:bob@contoso.com;opaque=user:epid:fvceImg6cFWG7W6lvUt_-AAA;gruu="">
User-Agent: UCCP/3.0.6789.0 OC/3.0.6789.0 (Microsoft Office Communicator)
Proxy-Authorization: …
Content-Type: application/msrtc-setDelegate +xml
Content-Length: …

<setDelegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegate-management"
version="4">
  <delegate uri="sip:john@contoso.com" action="remove"/>
</setDelegates>
```

The server sends a 200 OK response to bob@contoso.com, as follows:

```
SIP/2.0 200 OK
Authentication-Info: …
Content-Length: …
From: "Bob"<sip:bob@contoso.com>; tag=97c16ecf07;epid=0b196426d9
To: <sip:bob@contoso.com>;tag=77E194543CA4ACA978CF0109285FCC40
Call-ID: d4a50beb5058445cb0377a6c3692c42c
CSeq: 1 SERVICE
Via: SIP/2.0/TCP 172.24.32.124:53925;ms-received-port=53925;ms
received-cid=B00
Content-Type: application/vnd-microsoft-roaming-self+xml

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
 xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
 xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
 xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
 xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="5"/>
</roamingData>
```

The server then sends out a notification on all active self subscription dialogs, as follows:

```
BENOTIFY sip:172.24.32.124:53925;transport=tcp;ms-opaque=670c5ba27d;ms-received-cid=B00
SIP/2.0
Via: SIP/2.0/TCP 172.24.41.100;branch=z9hG4bK46F0BC36.0E24F0FE;branched=FALSE
Authentication-Info: …
Max-Forwards: 70
To: <sip:bob@contoso.com>;tag=486ec43e97;epid=84d3db8c23
Content-Length: …
From: <sip:bob@contoso.com>;tag=BE180080
Call-ID: 3703383eebdd4630905e81c9e4eb5e34
CSeq: 16 BENOTIFY
Require: eventlist
Content-Type: application/vnd-microsoft-roaming-self+xml
Event: vnd-microsoft-roaming-self
subscription-state: active;expires=46449

<roamingData xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
 xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
 xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
 xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
 xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <delegates xmlns="http://schemas.microsoft.com/2007/09/sip/delegates" version="5"/>
</roamingData>
```

4.13 Server Publication Category

The following examples illustrate the kinds of publication performed by the server.

contactCard

```
<category name="contactCard" instance="0" publishTime="2007-12-21T20:51:25.070"
container="0" version="15" expireType="static">
  <contactCard xmlns="http://schemas.microsoft.com/2006/09/sip/contactcard">
    <identity>
      <name>
        <displayName>Bob</displayName>
      </name>
      <email>bob@exchange.contoso.com</email>
    </identity>
  </contactCard>
</category>
```

Similar **contactCard** instances are generated for other containers as specified in the preceding table.

legacyInterop

```
<category name="legacyInterop" instance="0"
          publishTime="2007-12-21T20:51:25.070"
          container="32000" version="15" expireType="static">
  <legacyInterop availability="18500" />
</category>
```

userProperties

```
<category name="userProperties" instance="0"
          publishTime="2008-01-23T07:50:12.873"
          container="1" version="7" expireType="static">
  <userProperties>
    <lines>
      <line lineType="Uc">
        tel:+11234567890;ext=67890
      </line>
    </lines>
    <telephonyMode>Uc</telephonyMode>
    <facsimileTelephoneNumber>+11234567891</facsimileTelephoneNumber>
    <streetAddress>1 Anywhere Street</streetAddress>
    <l>Somewhere</l>
    <st>SomeState</st>
    <countryCode>US</countryCode>
    <postalCode>12345</postalCode>
    <wWWHomePage>http://contoso.com</wWWHomePage>
    <exumEnabled>1</exumEnabled>
    <exumURL>
      EUM:bob@contoso.com;phone-context=EX-OCS-SIPSec.exchange.contoso.com
    </exumURL>
  </userProperties>
</category>
```

mwi

```
<category name="mwi" instance="0"
     publishTime="2011-04-19T12:10:49.780"
     container="1" version="955" expireType="static">

  <mwi xmlns="http://schemas.microsoft.com/2006/09/sip/mwi"
```

```
    messageWaiting="true"
    unreadVoiceMailCount="33"
    readVoiceMailCount="24" />
</category>
```

4.14 Unified Contact Store

The following section contains the protocol examples of the request and responses from the server to start migration and when migration is completed.

4.14.1 Start Migration

The following shows the client nudging the server to start the migration. The ms-ucs-ready is the header that starts the migration.

Request to start migration:

```
SUBSCRIBE
Contact: < bob@contoso.com m;opaque=user:epid:hQ0SP 5rl16D4JvbBGGtAQAA;gruu>
User-Agent: UCCAPI/15.0.3419.3001 OC/15.0.3419.3003 (AppName)
Event: vnd-microsoft-roaming-contacts
Accept: application/vnd-microsoft-roaming-contacts+xml
Supported: ms-ucs
Supported: ms-ucs-ready
Supported: com.microsoft.autoextend
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="0A9339AD", targetname="exbox.exchange.corp.microsoft.com", crand="e8b69f85",
cnum="3", response="98ccabbd2f338e344bcf43657094492c4460b932"
Content-Length: 0
```

If the migration is allowed by an admin setting and has been triggered successfully the enhanced presence server will respond with ucsMode="allowed" as shown below:

```
SIP/2.0 200 OK
Contact: <sip:user5.contoso.com:5061;transport=tls;ms-fe=SERVER.contoso.com>
Proxy-Authentication-Info: Kerberos qop="auth", opaque="3ADB1284", srand="561A142B",
snum="3", rspauth="040401ffffffffff00000000000000037b82e69069d7b6ecf9a71f4",
targetname="sip/SERVER.contoso.com", realm="SIP Communications Service", version=4
Content-Length: 1239
From: "User 5"<sip:user5@contoso.com>;tag=435520b0a8;epid=a8716485d4
To: <sip: user5@contoso.com>;tag=C0B5AE6F
Call-ID: f9971f8004b34e7497573f3dbefe3caf
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 192.168.0.244:51649;ms-received-port=51649;ms-received-cid=82100
Expires: 30384
Content-Type: application/vnd-microsoft-roaming-contacts+xml
Event: vnd-microsoft-roaming-contacts
subscription-state: active;expires=30384
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

<contactList deltaNum="16" ucsMode="allowed" >
    <group id="1" name="Group 1 Name" externalURI=""  />
    <group id="2" name=" Group 2 Name " externalURI=""  />
    <contact uri="user0@contoso.com" name="" groups="1" subscribed="true"
        externalURI=""  />
    <contact uri="user1@contoso.com" name="" groups="1" subscribed="true"
        externalURI=""  />
    <contact uri="+14252057570@ contoso.com" name="" groups="1"
        subscribed="true" externalURI="" >
```

```
        <contactExtension><contactSettings></contactSettings>
              <phone><uri>tel:+14252057570</uri></phone>
         </contactExtension>
     </contact>
</contactList>
```

If the migration has not been triggered or is blocked by an Admin setting the enhanced presence server will respond with ucsMode="disabled" as shown below:

```
SIP/2.0 200 OK
Contact: <sip:user5.contoso.com:5061;transport=tls;ms-fe=SERVER.contoso.com>
Proxy-Authentication-Info: Kerberos qop="auth", opaque="3ADB1284", srand="561A142B",
snum="3", rspauth="040401ffffffffff00000000000000037b82e69069d7b6ecf9a71f4",
targetname="sip/SERVER.contoso.com", realm="SIP Communications Service", version=4
Content-Length: 1239
From: "User 5"<sip:user5@contoso.com>;tag=435520b0a8;epid=a8716485d4
To: <sip:user5@contoso.com>;tag=C0B5AE6F
Call-ID: f9971f8004b34e7497573f3dbefe3caf
CSeq: 1 SUBSCRIBE
Via: SIP/2.0/TLS 192.168.0.244:51649;ms-received-port=51649;ms-received-cid=82100
Expires: 30384
Content-Type: application/vnd-microsoft-roaming-contacts+xml
Event: vnd-microsoft-roaming-contacts
subscription-state: active;expires=30384
ms-piggyback-cseq: 1
Supported: ms-benotify, ms-piggyback-first-notify

<contactList deltaNum="16" ucsMode="disabled" >
    <group id="1" name="~" externalURI=""  />
    <group id="2" name="Group 1 Name" externalURI=""  />
    <group id="4" name=" Group 2 Name " externalURI=""  />
    <contact uri="user0@contoso.com" name="" groups="1" subscribed="true"
        externalURI=""  />
    <contact uri="user1@contoso.com" name="" groups="1" subscribed="true"
        externalURI=""  />
    <contact uri="+14252057570@ contoso.com" name="" groups="1"
        subscribed="true" externalURI="" >
        <contactExtension><contactSettings></contactSettings>
              <phone><uri>tel:+14252057570</uri></phone>
         </contactExtension>
     </contact>
</contactList>
```

4.14.2 On Migration Completed

If the SIP enabled end point is connected to the enhanced presence server when the Migration is completed, the end point will get the following notification:

2186; reason="Contact subscription has been terminated as the user migrated to ucs mode"

the Subscription will be terminated.

```
    BENOTIFY sip:192.168.0.244:51649;transport=tls;ms-opaque=2cea88f4f1;ms-received-
cid=82100;grid SIP/2.0
    Via: SIP/2.0/TLS
192.168.0.240:5061;branch=z9hG4bK8F129E46.313FFCD1C3E37643;branched=FALSE
    Authentication-Info: TLS-DSK qop="auth", opaque="4671A717", srand="605CF985",
snum="3", rspauth="25d341d7f62de8ebba153b7094d2821d434a7949",
targetname="SERVER.constoso.com", realm="SIP Communications Service", version=4
    Max-Forwards: 70
    To: <sip:user5@constoso.com>;tag=435520b0a8;epid=a8716485d4
    Content-Length: 1239
    From: <sip:user5@contoso.com>;tag=C0B5AE6F
    Call-ID: f9971f8004b34e7497573f3dbefe3caf
```

```
    CSeq: 3 BENOTIFY
    Content-Type: application/vnd-microsoft-roaming-contacts+xml
    Event: vnd-microsoft-roaming-contacts
    subscription-state: terminated;expires=0
    Expires: 0
    ms-diagnostics-public: 2186;reason="Contact subscription has been terminated as the
user migrated to ucs mode."

    <contactList deltaNum="16" ucsMode="migrated" >
        <group id="1" name="Group 1 Name" externalURI=""  />
        <group id="2" name=" Group 2 Name " externalURI=""  />
        <contact uri="user1@contoso.com" name="" groups="1" subscribed="true"
            externalURI=""  />
        <contact uri="user2@contoso.com" name="" groups="1" subscribed="true"
            externalURI=""  />
        <contact uri="+14252057570@contose.com" name="" groups="1"
            subscribed="true" externalURI="" >
            <contactExtension><contactSettings></contactSettings>
                <phone><uri>tel:+14252057570</uri></phone>
            </contactExtension>
        </contact>
    </contactList>
```

4.14.3 Post Migration

The following section shows the how endpoints can let the server know if they can connect to the email server and what the responses would be

4.14.3.1 Retrieving the Contact and Group List for Clients that can Connect to the Email Server

After migration has completed, SIP enabled end points that can connect to the email server first subscribes to the enhanced Presence Server to know if the contact list has been migrated with the "ms-ucs" supported header

```
  SUBSCRIBE
  Contact: < user5@contoso.com m;opaque=user:epid:hQ0SP_5rl16D4JvbBGGtAQAA;gruu>
  User-Agent: UCCAPI/15.0.3419.3001 OC/15.0.3419.3003 (AppName)
  Event: vnd-microsoft-roaming-contacts
  Accept: application/vnd-microsoft-roaming-contacts+xml
  Supported: ms-ucs
  Supported: com.microsoft.autoextend
  Supported: ms-benotify
  Proxy-Require: ms-benotify
  Supported: ms-piggyback-first-notify
  Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
  opaque="0A9339AD", targetname=" exbox.exchange.corp.microsoft.com", crand="e8b69f85",
  cnum="3", response="98ccabbd2f338e344bcf43657094492c4460b932"
  Content-Length: 0

Response from the enhanced Presence Server terminating the subscription with a 2165
diagnostic code:


  SIP/2.0 488 Not acceptable here
  Authentication-Info: TLS-DSK qop="auth", opaque="4671A717", srand="FA78E074", snum="5",
  rspauth="0f99e36e9009f0fff70a32c0524bd59cc5ae5cd9", targetname="SERVER.contoso.com",
  realm="SIP Communications Service", version=4
  From: "User 5"<sip:user5@contoso.com>;tag=6705176eed;epid=a8716485d4
  To: <sip:user5@contoso.com>;tag=DD687A6743502E056B92FABCF6DC7140
  Call-ID: 112629f5c5f346b09e647523d70ea03b
  CSeq: 1 SUBSCRIBE
  Via: SIP/2.0/TLS 192.168.0.244:51649;ms-received-port=51649;ms-received-cid=82100
```

```
ms-diagnostics: 2165;reason="Contact subscription is not allowed as the user's contact
list has migrated to Exchange.";source="SERVER.contoso.com"
Server: RTC/5.0
Content-Length: 0
```

4.14.3.2 Retrieving the Contact and Group List for Clients that cannot Connect to the Email Server

Request:

```
SUBSCRIBE
Contact: < bob@contoso.com m;opaque=user:epid:hQ0SP_5rl16D4JvbBGGtAQAA;gruu>
User-Agent: UCCAPI/15.0.3419.3001 OC/15.0.3419.3003 (AppName)
Event: vnd-microsoft-roaming-contacts
Accept: application/vnd-microsoft-roaming-contacts+xml
Supported: com.microsoft.autoextend
Supported: ms-benotify
Proxy-Require: ms-benotify
Supported: ms-piggyback-first-notify
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="0A9339AD", targetname="exbox.exchange.corp.microsoft.com", crand="e8b69f85",
cnum="3", response="98ccabbd2f338e344bcf43657094492c4460b932"
Content-Length: 0
```

Response will be the same as in the non-migrated mode.

5 Security

5.1 Security Considerations for Implementers

The extensions defined in this document do not require any special security considerations beyond what is natively defined for the Session Initiation Protocol (SIP).

5.2 Index of Security Parameters

None.

6 Appendix A: Enhanced Presence XML Schemas

References in this section to rich preference are not related to [RFC4480], which defines "Rich Presence Extensions to the Presence Information Data Format". In this section, "rich presence" and "enhanced presence" refer to the enhanced presence model and are used interchangeably.

6.1 Common Schemas

Location: http://schemas.microsoft.com/2006/09/sip/rich-presence-common

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="ContainerId">
    <xs:restriction base="xs:unsignedShort">
      <xs:maxInclusive value="32767"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ContainerIdForSetContainer">
    <xs:restriction base="xs:unsignedShort">
      <!--
           Container 0 is not meant to be modified. It is the default container
           with assumed membership, and logically has no membership list.
       -->
      <xs:minExclusive value="0"/>
      <xs:maxInclusive value="32767"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

6.2 Categories Document

Location: http://schemas.microsoft.com/2006/09/sip/categories

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/categories"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:rpc="http://schemas.microsoft.com/2006/09/sip/rich-presence-common">
  <xs:import
    namespace="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
    schemaLocation="RichPresenceCommon.xsd"/>
  <xs:complexType name="CategoryType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="instance" type="xs:unsignedInt" use="optional"/>
    <xs:attribute name="publishTime" type="xs:string" use="optional"/>
    <xs:attribute name="container" type="rpc:ContainerId" use="optional"/>
    <xs:attribute name="version" type="xs:unsignedInt" use="optional"/>
    <xs:attribute name="expireType" use="optional">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="static"/>
          <xs:enumeration value="endpoint"/>
          <xs:enumeration value="user"/>
          <xs:enumeration value="time"/>
        </xs:restriction>
      </xs:simpleType>
```

```
    </xs:attribute>
    <xs:attribute name="endpointId" type="xs:string" use="optional"/>
    <xs:attribute name="expires" type="xs:integer" use="optional"/>
  </xs:complexType>
  <xs:complexType name="CategoriesType">
    <xs:sequence>
      <xs:element name="category" type="CategoryType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="uri" type="xs:anyURI" use="required"/>
  </xs:complexType>
  <xs:element name="categories" type="CategoriesType"/>
</xs:schema>
```

6.3 Containers Document

Location: http://schemas.microsoft.com/2006/09/sip/containers

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/containers"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:rpc="http://schemas.microsoft.com/2006/09/sip/rich-presence-common">
  <xs:import
    namespace="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
    schemaLocation="RichPresenceCommon.xsd"/>
  <xs:complexType name="ContainerMemberType">
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="user"/>
          <xs:enumeration value="domain"/>
          <xs:enumeration value="sameEnterprise"/>
          <xs:enumeration value="federated"/>
          <xs:enumeration value="publicCloud"/>
          <xs:enumeration value="everyone"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="value" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="ContainerType">
    <xs:sequence>
      <xs:element name="member" type="ContainerMemberType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="rpc:ContainerId" use="required"/>
    <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
  </xs:complexType>
  <xs:complexType name="ContainersType">
    <xs:sequence>
      <xs:element name="container" type="ContainerType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="containers" type="ContainersType"/>
</xs:schema>
```

6.4 Subscriber List Document

Location: http://schemas.microsoft.com/2006/09/sip/presence-subscribers

```
<xs:schema
```

```
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="AckSubscriberType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="user" type="xs:string" use="required"/>
    <xs:attribute name="acknowledged" type="xs:boolean" use="required" fixed="true"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="ContextType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="SubscriberType">
    <xs:sequence>
      <xs:element name="context" minOccurs="0" maxOccurs="1" type="ContextType"/>
    </xs:sequence>
    <xs:attribute name="user" type="xs:string" use="required"/>
    <xs:attribute name="displayName" type="xs:string" use="required"/>
    <xs:attribute name="acknowledged" type="xs:boolean" use="required"/>
    <xs:attribute name="type" use="optional">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="federated"/>
          <xs:enumeration value="publicCloud"/>
          <xs:enumeration value="sameEnterprise"/>
          <xs:enumeration value="unknown"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="SubscribersType">
    <xs:sequence>
      <xs:element name="subscriber" type="SubscriberType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="subscribers" type="SubscribersType"/>
</xs:schema>
```

6.5 Delegates Document

Location: http://schemas.microsoft.com/2007/09/sip/delegates

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2007/09/sip/delegates"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2007/09/sip/delegates"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="DelegateType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
      <xs:attribute name="uri" type="xs:anyURI" use="required"/>
      <xs:attribute name="publish" type="xs:boolean" use="required"/>
      <xs:attribute name="redelegate" type="xs:boolean" use="required"/>
      <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
  <xs:complexType name="DelegatesType">
    <xs:sequence>
```

```
      <xs:element name="delegate" type="DelegateType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
    <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
  <xs:element name="delegates" type="DelegatesType"/>
</xs:schema>
```

6.6 Self SUBSCRIBE Request

Location: http://schemas.microsoft.com/2007/09/sip/roaming-self-ex

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2007/09/sip/roaming-self-ex"
    elementFormDefault="qualified"
    xmlns="http://schemas.microsoft.com/2007/09/sip/roaming-self-ex"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="roamingTypeEnumEx">
    <xs:restriction base="xs:string">
      <xs:enumeration value="delegates"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="roamingTypeUnion">
    <xs:union memberTypes="roamingTypeEnumEx xs:string"/>
  </xs:simpleType>
  <xs:complexType name="RoamingTypeEx">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="type" use="required" type="roamingTypeUnion"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="roamingEx" type="RoamingTypeEx"/>
</xs:schema>
```

Location: http://schemas.microsoft.com/2006/09/sip/roaming-self

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    elementFormDefault="qualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:rex="http://schemas.microsoft.com/2007/09/sip/roaming-self-ex"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="presenceSeparator">
    <xs:complexType/>
  </xs:element>
  <xs:complexType name="RoamingType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="categories"/>
          <xs:enumeration value="containers"/>
          <xs:enumeration value="subscribers"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
```

```
  <xs:complexType name="RoamingListType">
    <xs:sequence>
      <xs:element name="roaming" type="RoamingType" maxOccurs="unbounded"/>
   <xs:element ref="rex:roamingEx" minOccurs="0" maxOccurs="unbounded"/>
        <xs:sequence minOccurs="0">
          <xs:element ref="presenceSeparator"/>
          <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
        </xs:sequence>
        </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="roamingList" type="RoamingListType"/>
</xs:schema>
```

6.7 Self SUBSCRIBE Response

Location: http://schemas.microsoft.com/2006/09/sip/roaming-self

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/roaming-self"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:cat="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:con="http://schemas.microsoft.com/2006/09/sip/containers"
    xmlns:sub="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
    xmlns:del="http://schemas.microsoft.com/2007/09/sip/delegates">
  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/categories"
      schemaLocation="CategoriesNotification.xsd"/>
  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
      schemaLocation="SubscribersNotification.xsd"/>
  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/containers"
      schemaLocation="ContainersNotification.xsd"/>
  <xs:import
      namespace="http://schemas.microsoft.com/2007/09/sip/delegates"
      schemaLocation="DelegatesNotification.xsd"/>
  <xs:complexType name="RoamingDataType">
    <xs:sequence>
      <xs:element ref="cat:categories" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="con:containers" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="sub:subscribers" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="del:delegates" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="roamingData" type="RoamingDataType"/>
</xs:schema>
```

6.8 Batch Categories SUBSCRIBE Request

Location: http://schemas.microsoft.com/2006/09/sip/categorylist

Note: http://schemas.microsoft.com/2006/01/sip/batch-subscribe refers to http://schemas.microsoft.com/2006/09/sip/categorylist.

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/categorylist"
    elementFormDefault="qualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/categorylist"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="CategoryType">
```

```
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="CategoryListType">
    <xs:sequence>
      <xs:element name="category" type="CategoryType" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="categoryList" type="CategoryListType"/>
</xs:schema>
```

Location: http://schemas.microsoft.com/2006/01/sip/batch-subscribe

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
    elementFormDefault="qualified"
    xmlns="http://schemas.microsoft.com/2006/01/sip/batch-subscribe"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:cl="http://schemas.microsoft.com/2006/09/sip/categorylist">
  <xs:import
    namespace="http://schemas.microsoft.com/2006/09/sip/categorylist"
    schemaLocation="categoryList.xsd"/>
  <xs:complexType name="ContextType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ResourceType">
    <xs:sequence>
      <xs:element name="context" minOccurs="0" maxOccurs="1" type="ContextType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="uri" type="xs:anyURI" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="AdhocListType">
    <xs:sequence>
      <xs:element name="resource" type="ResourceType" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ActionType">
    <xs:sequence>
      <xs:element name="adhocList" type="AdhocListType" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="cl:categoryList" minOccurs="1" maxOccurs="1"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="subscribe"/>
          <xs:enumeration value="unsubscribe"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
```

```
  <xs:element name="batchSub">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="action" type="ActionType" maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

6.9 Batch Categories SUBSCRIBE Response

```
<xs:schema
    targetNamespace="urn:ietf:params:xml:ns:rlmi"
    elementFormDefault="qualified" xmlns="urn:ietf:params:xml:ns:rlmi"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="list">
    <xs:sequence>
      <xs:element ref="name" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="resource" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="uri" type="xs:anyURI" use="required"/>
    <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
    <xs:attribute name="fullState" type="xs:boolean" use="required"/>
    <xs:attribute name="cid" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:element name="list" type="list"/>
  <xs:element name="resource">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="name" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="instance" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="uri" type="xs:anyURI" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="instance">
    <xs:complexType>
      <xs:attribute name="id" type="xs:unsignedInt" use="required"/>
      <xs:attribute name="state" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="active"/>
            <xs:enumeration value="pending"/>
            <xs:enumeration value="terminated"/>
            <xs:enumeration value="resubscribe"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="reason" type="xs:string" use="optional"/>
      <xs:attribute name="statusCode" type="xs:unsignedInt" use="optional"/>
      <xs:attribute name="cid" type="xs:string" use="optional"/>
      <xs:attribute name="poolFqdn" type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="name">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="language" type="xs:string" use="optional"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
```

```
</xs:schema>
```

6.10 Category Publish Request

Location: http://schemas.microsoft.com/2006/09/sip/rich-presence

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/rich-presence"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/rich-presence"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:rpc="http://schemas.microsoft.com/2006/09/sip/rich-presence-common">
  <xs:import
    namespace="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
    schemaLocation="RichPresenceCommon.xsd"/>
  <xs:complexType name="PublicationType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="1">
      </xs:any>
    </xs:sequence>
    <xs:attribute name="categoryName" type="xs:string" use="required"/>
    <xs:attribute name="instance" type="xs:unsignedInt" use="required"/>
    <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
    <xs:attribute name="container" type="rpc:ContainerId" use="required"/>
    <xs:attribute name="expireType" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="static"/>
         <xs:enumeration value="endpoint"/>
         <xs:enumeration value="user"/>
         <xs:enumeration value="time"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="expires" type="xs:unsignedInt" use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="PublicationsType">
    <xs:sequence>
      <xs:element name="publication" minOccurs="1" maxOccurs="unbounded"
type="PublicationType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="uri" type="xs:anyURI" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="PublishType">
    <xs:sequence>
      <xs:element name="publications" minOccurs="1" maxOccurs="1" type="PublicationsType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="publish" type="PublishType"/>
</xs:schema>
```

6.11 setContainerMember Request

Location: http://schemas.microsoft.com/2006/09/sip/container-management

```
<xs:schema
```

```
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/container-management"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2006/09/sip/container-management"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:rpc="http://schemas.microsoft.com/2006/09/sip/rich-presence-common">
  <xs:import namespace="http://schemas.microsoft.com/2006/09/sip/rich-presence-common"
schemaLocation="RichPresenceCommon.xsd"/>
  <xs:complexType name="ContainerMemberType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="user"/>
          <xs:enumeration value="domain"/>
          <xs:enumeration value="sameEnterprise"/>
          <xs:enumeration value="federated"/>
          <xs:enumeration value="publicCloud"/>
          <xs:enumeration value="everyone"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="value" type="xs:string" use="optional"/>
    <xs:attribute name="action" use="optional" default="add">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="add"/>
          <xs:enumeration value="remove"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="ContainerType">
    <xs:sequence>
      <xs:element name="member" type="ContainerMemberType" minOccurs="1"
maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="rpc:ContainerIdForSetContainer" use="required"/>
    <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="SetContainerMembersType">
    <xs:sequence>
      <xs:element name="container" type="ContainerType" minOccurs="1"
maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="setContainerMembers" type="SetContainerMembersType"/>
</xs:schema>
```

6.12 setSubscriber Request

Location: http://schemas.microsoft.com/2006/09/sip/presence-subscribers

```
  <?xml version="1.0" encoding="UTF-8"?>
  <xs:schema xmlns="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
             targetNamespace="http://schemas.microsoft.com/2006/09/sip/presence-subscribers"
```

```
            elementFormDefault="qualified"
            attributeFormDefault="unqualified"
            xmlns:xs="http://www.w3.org/2001/XMLSchema"
            xmlns:ms="urn:microsoft-cpp-xml-serializer">
    <xs:complexType name="AckSubscriberType" ms:className="CSubscriberXmlDocumentResult">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="user" type="xs:string" use="required" ms:propertyName="User"/>
        <xs:attribute name="acknowledged" type="xs:boolean" use="required" fixed="true"
ms:propertyName="Acknowledged"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
    <xs:complexType name="SetSubscribersType"
ms:className="CSetSubscribersXmlDocumentResult">
        <xs:sequence>
            <xs:element name="subscriber" type="AckSubscriberType" minOccurs="1"
maxOccurs="1" ms:propertyName="Subscriber"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
    <xs:element name="setSubscribers" type="SetSubscribersType"
ms:className="CSetSubscribersXmlDocumentRoot"/>
</xs:schema>
```

6.13 SubscriptionContext

Location: http://schemas.microsoft.com/2008/09/sip/subscription-context

```
<xs:schema
  targetNamespace="http://schemas.microsoft.com/2008/09/sip/SubscriptionContext"
  xmlns:tns="http://schemas.microsoft.com/2008/09/sip/SubscriptionContext"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
  attributeFormDefault="unqualified"
  elementFormDefault="qualified">

  <xs:import namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
             schemaLocation="commontypes.xsd"/>
  <xs:complexType name="contactListType">
    <xs:sequence>
      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter" />
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:element ref="ct:end" />
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
    </xs:complexType>

  <xs:complexType name="watcherType">
    <xs:sequence>
      <!-- Recipient is part of watcher's contact list.-->
      <xs:element name="contactList" type="tns:contactListType"/>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter" />
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
```

```
        </xs:sequence>
        <xs:element ref="ct:end" />
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1" />

    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="delegatorType">
    <xs:sequence>
      <xs:element name="delegator" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter" />
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:element ref="ct:end" />
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="subscriptionContextType">
    <xs:sequence>

      <!-- Recipient will be watched by a watcher.-->
      <xs:element name="watcher" type="tns:watcherType" minOccurs="0" maxOccurs="1"/>

      <!-- Recipient is a delegate. -->
      <xs:element name="delegation" type="tns:delegatorType" minOccurs="0" maxOccurs="1"/>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter" />
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:element ref="ct:end" />
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1" />


    </xs:sequence>

    <!--
    Protocol client does not proceed with the parsing if its major version doesn't
    match to the version in the xml.
    Change in major version indicates that format is not compatible.
    -->
    <xs:attribute name="majorVersion" type="xs:unsignedInt" use="required" />

    <!--
    Change in minor version without any change in the major version indicates that
    format is backward compatible.
    -->
    <xs:attribute name="minorVersion" type="xs:unsignedInt" use="required" />

    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>

  <xs:element name="subscriptionContext" type="tns:subscriptionContextType"/>
</xs:schema>
```

6.14 Delegation Request

Location: http://schemas.microsoft.com/2007/09/sip/delegate-management

```
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2007/09/sip/delegate-management"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    xmlns="http://schemas.microsoft.com/2007/09/sip/delegate-management"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="setDelegates">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="delegate" type="DelegateElement"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:unsignedInt" use="required"/>
      <xs:anyAttribute namespace="##other"/>
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="authorizationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="allow"/>
      <xs:enumeration value="deny"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="PublishDelegateOperationType">
    <xs:attribute name="authorization" type="authorizationType" use="required"/>
    <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
  <xs:complexType name="RedelegateDelegateOperationType">
    <xs:attribute name="authorization" type="authorizationType" use="required"/>
    <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
  <xs:complexType name="DelegateElement">
    <xs:sequence>
      <xs:element name="publishOperation" type="PublishDelegateOperationType"
minOccurs="0"/>
      <xs:element name="redelegateOperation" type="RedelegateDelegateOperationType"
minOccurs="0"/>
      <xs:any namespace="##other" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="uri" type="xs:anyURI" use="required"/>
    <xs:attribute name="action" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="add"/>
          <xs:enumeration value="remove"/>
          <xs:enumeration value="modify"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:anyAttribute namespace="##other"/>
  </xs:complexType>
</xs:schema>
```

7 Appendix B: Individual Category XML Schemas

7.1 Common Schemas

Location: http://schemas.microsoft.com/2006/09/sip/commontypes

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
    xmlns:tns="http://schemas.microsoft.com/2006/09/sip/commontypes"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:element name="delimiter">
    <xs:complexType />
  </xs:element>

  <xs:element name="end">
    <xs:complexType />
  </xs:element>

  <xs:complexType name="extensionType">
    <xs:sequence>
      <xs:any processContents="lax" minOccurs="1"
          maxOccurs="unbounded" namespace="##other"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="extension" type="tns:extensionType"/>

</xs:schema>
```

Location http://schemas.microsoft.com/2010/12/sip/propertyList

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://schemas.microsoft.com/2010/12/sip/propertyList"
  xmlns:tns="http://schemas.microsoft.com/2010/12/sip/propertyList"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <xs:import namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
             schemaLocation="commontypes.xsd"/>

  <!-- Type definitions -->

  <xs:simpleType name="typeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="boolean" />
      <xs:enumeration value="numeric" />
      <xs:enumeration value="string" />
      <xs:enumeration value="object" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="typeEnumEx">
    <xs:union memberTypes="tns:typeEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="propertyType">
    <xs:choice>
```

```
      <xs:sequence minOccurs="1" maxOccurs="unbounded">
        <xs:element name="property" type="tns:propertyType" />
      </xs:sequence>
      <xs:element name="value" type="xs:string" />
    </xs:choice>
    <xs:attribute name="name" type="xs:string" use="required" />
    <xs:attribute name="type" type="tns:typeEnumEx" use="required" />
    <xs:anyAttribute processContents="lax" />
  </xs:complexType>

  <xs:complexType name="propertyListType">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="property" type="tns:propertyType" />
      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax"
            minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>

  <!-- Schema elements -->

  <xs:element name="propertyList" type="tns:propertyListType" />

</xs:schema>
```

7.2 state/dndState Category

Location: http://schemas.microsoft.com/2006/09/sip/state

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://schemas.microsoft.com/2006/09/sip/state"
  xmlns:tns="http://schemas.microsoft.com/2006/09/sip/state"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  >

  <xs:include schemaLocation="statetypes.xsd"/>

  <xs:import namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
             schemaLocation="commontypes.xsd"/>

  <xs:annotation>
    <xs:documentation>
      This Schema identifies state.
    </xs:documentation>
  </xs:annotation>

  <xs:simpleType name="deviceTypeEnum">
    <xs:annotation>
      <xs:documentation>
        computer:  Office communicator.
        deskphone: Office communicator phone edition, a.k.a. DOMO/Tanjay.
        mobile:    Office communicator mobile edition, a.k.a. COMO.
        web:       Office communicator web access, a.k.a. CWA.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:token">
```

```
      <xs:enumeration value="computer"/>
      <xs:enumeration value="deskphone"/>
      <xs:enumeration value="mobile"/>
      <xs:enumeration value="web"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="deviceTypeEnumEx">
    <xs:union memberTypes="tns:deviceTypeEnum xs:token" />
  </xs:simpleType>

  <xs:simpleType name="endpointLocationEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="office"/>
      <xs:enumeration value="mobile"/>
      <xs:enumeration value="home"/>
      <xs:enumeration value="none"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="endpointLocationEnumEx">
    <xs:union memberTypes="tns:endpointLocationEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="extensionType">
    <xs:sequence>
      <xs:any processContents="lax" minOccurs="1"
              maxOccurs="unbounded" namespace="##targetNamespace"/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="unboundedType">
    <xs:union memberTypes="xs:unsignedInt">
      <xs:simpleType>
        <xs:restriction base="xs:token">
          <xs:enumeration value="unbounded"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:union>
  </xs:simpleType>

  <xs:simpleType name="activityTokenTypeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="on-the-phone"/>
      <xs:enumeration value="in-a-conference"/>
      <xs:enumeration value="in-a-meeting"/>
      <xs:enumeration value="out-of-office"/>
      <xs:enumeration value="urgent-interruptions-only"/>
      <xs:enumeration value="off-work"/>
      <xs:enumeration value="in-presentation"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="activityTokenTypeEnumEx">
    <xs:union memberTypes="tns:activityTokenTypeEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="activityType">
    <xs:sequence>
      <xs:element name="custom" type="tns:LCIDType" minOccurs="0" maxOccurs="unbounded"/>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
```

```
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>


    </xs:sequence>
    <xs:attribute name="token" type="tns:activityTokenTypeEnumEx" use="optional"/>
    <xs:attribute name="maxAvailability" type="tns:unboundedType" use="optional"
default="unbounded"/>
    <xs:attribute name="minAvailability" type="tns:unboundedType" use="optional"
default="0"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="stateType" abstract="true">
    <xs:sequence>
      <xs:element name="availability" type="xs:unsignedInt" minOccurs="0"/>
      <xs:element name="activity" type="tns:activityType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="endpointLocation" type="tns:endpointLocationEnumEx" minOccurs="0"/>

      <xs:element name="extension"  type="tns:extensionType" minOccurs="0" maxOccurs="1"/>

    </xs:sequence>
    <xs:attribute name="manual" type="xs:boolean" use="optional" default="false"/>
    <xs:attribute name="startTime" type="xs:dateTime" use="optional"/>
    <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
    <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="aggregateState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">
        <xs:sequence>
          <xs:element name="meetingSubject" type="tns:LCIDType" minOccurs="0"/>
          <xs:element name="meetingLocation" type="tns:LCIDType" minOccurs="0"/>

          <xs:sequence minOccurs="0" maxOccurs="1">

            <!-- W13 extension -->
            <xs:element ref="ct:delimiter"/>
            <xs:element name="timeZoneBias" type="xs:long" minOccurs="0"/>
            <xs:element name="timeZoneName" type="xs:string" minOccurs="0"/>
            <xs:element name="timeZoneAbbreviation" type="xs:string" minOccurs="0"/>
            <xs:element name="device" type="tns:deviceTypeEnumEx" minOccurs="0"
maxOccurs="1"/>

            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
        <xs:attribute name="lastActive" type="xs:dateTime" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="aggregateMachineState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">

        <xs:sequence>

          <xs:sequence minOccurs="0" maxOccurs="1">

            <!-- W13 extension -->
            <xs:element ref="ct:delimiter"/>
            <xs:element name="timeZoneBias" type="xs:long" minOccurs="0"/>
```

```
            <xs:element name="timeZoneName" type="xs:string" minOccurs="0"/>
            <xs:element name="timeZoneAbbreviation" type="xs:string" minOccurs="0"/>
            <xs:element name="device" type="tns:deviceTypeEnumEx" minOccurs="0"
maxOccurs="1"/>

            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>

        <xs:attribute name="endpointId" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="userState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">

        <xs:sequence>

          <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>

      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="presentingState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">
         <xs:sequence>
           <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>
         </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="machineState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">
        <xs:sequence>

          <xs:sequence minOccurs="0" maxOccurs="1">

            <!-- W13 extension -->
            <xs:element ref="ct:delimiter"/>
            <xs:element name="timeZoneBias" type="xs:long" minOccurs="0"/>
            <xs:element name="timeZoneName" type="xs:string" minOccurs="0"/>
```

```
            <xs:element name="timeZoneAbbreviation" type="xs:string" minOccurs="0"/>
            <xs:element name="device" type="tns:deviceTypeEnumEx" minOccurs="0"
maxOccurs="1"/>

            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <!-- W15 extension -->
              <xs:element ref="ct:delimiter"/>
              <xs:element name="treatLocationAsProximate" type="xs:string" minOccurs="0"/>
              <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element ref="ct:delimiter"/>
                <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
              </xs:sequence>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="phoneState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">

        <xs:sequence>

          <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>

        <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="calendarState">
    <xs:complexContent>
      <xs:extension base="tns:stateType">
        <xs:sequence>
          <xs:element name="meetingSubject" type="tns:LCIDType" minOccurs="0"/>
          <xs:element name="meetingLocation" type="tns:LCIDType" minOccurs="0"/>

          <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
        <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:element name="state" type="tns:stateType"/>
```

```
</xs:schema>
```

7.3 contactCard Category

Location: http://schemas.microsoft.com/2006/09/sip/contactcard

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/contactcard"
    xmlns:tns="http://schemas.microsoft.com/2006/09/sip/contactcard"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
      schemaLocation="commontypes.xsd"/>

  <xs:simpleType name="photoTypeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="default" />
      <xs:enumeration value="enterprise" />
      <xs:enumeration value="other" />
      <xs:enumeration value="exchange" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="photoTypeEnumEx">
    <xs:union memberTypes="tns:photoTypeEnum xs:token" />
  </xs:simpleType>

  <xs:simpleType name="phoneTypeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="work"/>
      <xs:enumeration value="home"/>
      <xs:enumeration value="mobile"/>
      <xs:enumeration value="other"/>
      <xs:enumeration value="custom1"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="phoneTypeEnumEx">
    <xs:union memberTypes="tns:phoneTypeEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="updatedType" abstract="true">
    <xs:attribute name="updated" type="xs:dateTime" use="optional"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>

  <xs:complexType name="updatedAnyURIType">
    <xs:simpleContent>
      <xs:extension base="xs:anyURI">
        <xs:attribute name="updated" type="xs:dateTime" use="optional"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="updatedStringType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="updated" type="xs:dateTime" use="optional"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
```

```
      </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="LCIDType">
      <xs:simpleContent>
        <xs:extension base="tns:updatedStringType">
          <xs:attribute name="LCID" type="xs:unsignedInt" use="optional"/>
          <xs:anyAttribute processContents="lax"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:simpleType name="urlTypeEnum">
      <xs:restriction base="xs:token">
        <xs:enumeration value="sharepoint"/>
        <xs:enumeration value="voicemail"/>
        <xs:enumeration value="other"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="urlTypeEnumEx">
      <xs:union memberTypes="tns:urlTypeEnum xs:token" />
    </xs:simpleType>

    <xs:complexType name="urlType">
      <xs:simpleContent>
        <xs:extension base="tns:updatedAnyURIType">
          <xs:attribute name="type" type="tns:urlTypeEnumEx"/>
          <xs:anyAttribute processContents="lax"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="photoType">
      <xs:complexContent>
        <xs:extension base="tns:updatedType">
          <xs:sequence>
            <xs:element name="uri" type="tns:updatedAnyURIType" minOccurs="0" maxOccurs="1"
  />
            <xs:element name="hash" type="xs:string" minOccurs="1" maxOccurs="1" />
            <xs:sequence minOccurs="0" maxOccurs="1">
              <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element ref="ct:delimiter" />
                <xs:any namespace="##targetNamespace" processContents="lax"
                     minOccurs="0" maxOccurs="unbounded" />
              </xs:sequence>
              <xs:element ref="ct:end" />
            </xs:sequence>
            <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1" />
          </xs:sequence>
          <xs:attribute name="type" type="tns:photoTypeEnumEx" use="required"/>
          <xs:anyAttribute processContents="lax" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="phoneType">
      <xs:complexContent>
        <xs:extension base="tns:updatedType">
          <xs:sequence>
            <xs:element name="uri" type="tns:updatedAnyURIType"
                        minOccurs="1" maxOccurs="1"/>
            <xs:element name="displayString" type="tns:LCIDType"
                        minOccurs="0" maxOccurs="unbounded"/>

            <xs:sequence minOccurs="0" maxOccurs="1">
              <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element ref="ct:delimiter"/>
                <xs:any namespace="##targetNamespace" processContents="lax"
```

```
                    minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
        <xs:attribute name="type" type="tns:phoneTypeEnumEx"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="nameType">
    <xs:complexContent>
      <xs:extension base="tns:updatedType">
        <xs:sequence>
          <xs:element name="displayName" type="tns:LCIDType"
                      minOccurs="0" maxOccurs="unbounded"/>

          <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax"
                      minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="identityType">
    <xs:complexContent>
      <xs:extension base="tns:updatedType">
        <xs:sequence>
          <xs:element name="name" type="tns:nameType" minOccurs="0"/>
          <xs:element name="email" type="tns:updatedAnyURIType"
                      minOccurs="0" maxOccurs="unbounded"/>

          <xs:sequence minOccurs="0" maxOccurs="1">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="ct:delimiter"/>
              <xs:any namespace="##targetNamespace" processContents="lax"
                      minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:simpleType name="addressTypeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="work"/>
      <xs:enumeration value="home"/>
      <xs:enumeration value="other"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="addressTypeEnumEx">
    <xs:union memberTypes="tns:addressTypeEnum xs:token" />
  </xs:simpleType>
```

```
<xs:complexType name="addressType">
  <xs:sequence>
    <xs:element name="street" type="tns:LCIDType"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="city" type="tns:LCIDType"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="state" type="tns:LCIDType"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="zipcode" type="tns:LCIDType"
                minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="countryCode" type="xs:token" minOccurs="0" maxOccurs="1"/>

    <xs:sequence minOccurs="0" maxOccurs="1">
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="ct:delimiter"/>
        <xs:any namespace="##targetNamespace" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:element ref="ct:end"/>
    </xs:sequence>
    <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

  </xs:sequence>
  <xs:attribute name="type" type="tns:addressTypeEnumEx"/>
  <xs:anyAttribute processContents="lax"/>
</xs:complexType>

<xs:simpleType name="presentityDescriptionType">
  <xs:restriction base="xs:string">
    <xs:maxLength value="1024" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="presentityTypeEnum">
  <xs:restriction base="xs:token">
    <xs:enumeration value="person" />
    <xs:enumeration value="huntgroup" />
    <xs:enumeration value="autoattendant" />
    <xs:enumeration value="automaton" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="presentityTypeEnumEx">
  <xs:union memberTypes="tns:presentityTypeEnum xs:token" />
</xs:simpleType>

<xs:complexType name="contactCardType">
  <xs:complexContent>
    <xs:extension base="tns:updatedType">
      <xs:sequence>
        <xs:element name="identity" type="tns:identityType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="address" type="tns:addressType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="company" type="tns:LCIDType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="department" type="tns:LCIDType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="title" type="tns:LCIDType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="office" type="tns:LCIDType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="url" type="tns:urlType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="phone" type="tns:phoneType" minOccurs="0"
                    maxOccurs="unbounded"/>
        <xs:element name="automaton" type="xs:boolean" minOccurs="0"
                    maxOccurs="1" default="false" />

        <xs:sequence minOccurs="0" maxOccurs="1">
```

```
            <xs:element ref="ct:delimiter" />
            <xs:element name="type" type="tns:presentityTypeEnumEx"
               minOccurs="0" maxOccurs="1" />
            <xs:element name="description"
               type="tns:presentityDescriptionType" minOccurs="0"
               maxOccurs="1" />
            <xs:sequence minOccurs="0" maxOccurs="1">
                <xs:element ref="ct:delimiter" />
                <xs:element name="displayADPhoto" type="xs:boolean"
                  minOccurs="0" maxOccurs="1" />
                <xs:element name="photo" type="tns:photoType" minOccurs="0"
                    maxOccurs="unbounded" />
                <xs:sequence minOccurs="0" maxOccurs="unbounded">
                  <xs:element ref="ct:delimiter" />
                  <xs:any namespace="##targetNamespace"
                       processContents="lax" minOccurs="0"
                       maxOccurs="unbounded" />
                </xs:sequence>
            </xs:sequence>
            <xs:element ref="ct:end"/>
          </xs:sequence>
          <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

        </xs:sequence>
        <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
        <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:element name="contactCard" type="tns:contactCardType"/>

</xs:schema>
```

7.4 device Category

Location: http://schemas.microsoft.com/2006/09/sip/device

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/device"
    xmlns:tns="http://schemas.microsoft.com/2006/09/sip/device"
    xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
      schemaLocation="commontypes.xsd"/>

  <xs:complexType name="preferredType" abstract="true">
    <xs:attribute name="preferred" type="xs:boolean" use="optional" default="false"/>
  </xs:complexType>

  <xs:complexType name="preferredEndpointType">
    <xs:complexContent>
      <xs:extension base="tns:preferredType">
        <xs:attribute name="preferredEndpointId" type="xs:string" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="capabilityType">
    <xs:complexContent>
```

```
      <xs:extension base="tns:preferredEndpointType">
        <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
        <xs:attribute name="render" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="capture" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="publish" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="version" type="xs:unsignedInt"
                      use="optional" default="0"/>
        <xs:attribute name="deviceAvailability" type="xs:unsignedInt"
                      use="optional"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="capabilitiesType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="calendar" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="remoteCallControl" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="voice" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="video" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="CCCP" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="text" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="gifInk" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="isfInk" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="breakthrough" type="tns:capabilityType"
               maxOccurs="1"/>
        <xs:element name="applicationSharing" type="tns:capabilityType"
            maxOccurs="1"/>
        <xs:element name="ucs" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="containerIntegrity" type="tns:capabilityType"
            maxOccurs="1"/>
        <xs:element name="contentWhiteboard" type="tns:capabilityType"
             maxOccurs="1"/>
        <xs:element name="contentPoll" type="tns:capabilityType"
             maxOccurs="1"/>
        <xs:element name="contentPowerPoint" type="tns:capabilityType"
        maxOccurs="1"/>
        <xs:element name="contentNativeFile" type="tns:capabilityType"
       maxOccurs="1"/>
        <xs:element name="contentSharedNotes" type="tns:capabilityType"
       maxOccurs="1"/>
      </xs:choice>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax"
                  minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

    </xs:sequence>

    <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
    <xs:attribute name="preferred" type="xs:boolean" use="optional"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>

  <xs:complexType name="deviceType">
    <xs:sequence>
      <xs:element name="capabilities" type="tns:capabilitiesType"
                  minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="timezone" type="xs:time" />
```

```
      <xs:element name="machineName" type="xs:string" />

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax"
                  minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

    </xs:sequence>
    <xs:attribute name="endpointId" use="required" type="xs:string"/>

    <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
    <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
    <xs:anyAttribute processContents="lax"/>

  </xs:complexType>
  <xs:element name="device" type="tns:deviceType"/>
</xs:schema>
```

7.5 services Category

Location: http://schemas.microsoft.com/2006/09/sip/service

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/service"
    xmlns:tns="http://schemas.microsoft.com/2006/09/sip/service"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:import
      namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
      schemaLocation="commontypes.xsd"/>

  <xs:complexType name="preferredType" abstract="true">
    <xs:attribute name="preferred" type="xs:boolean"
                  use="optional" default="false"/>
  </xs:complexType>

  <xs:complexType name="preferredEndpointType">
    <xs:complexContent>
      <xs:extension base="tns:preferredType">
        <xs:attribute name="preferredEndpointId" type="xs:string"
                      use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="capabilityType">
    <xs:complexContent>
      <xs:extension base="tns:preferredEndpointType">
        <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
        <xs:attribute name="render" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="capture" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="publish" type="xs:boolean"
                      use="optional" default="false"/>
        <xs:attribute name="version" type="xs:unsignedInt"
                      use="optional" default="0"/>
```

```
        <xs:attribute name="deviceAvailability" type="xs:unsignedInt"
                      use="optional"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="capabilitiesType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="calendar" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="remoteCallControl" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="voice" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="video" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="CCCP" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="text" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="gifInk" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="isfInk" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="breakthrough" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="applicationSharing" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="ucs" type="tns:capabilityType" maxOccurs="1"/>
        <xs:element name="containerIntegrity" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="contentWhiteboard" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="contentPoll" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="contentPowerPoint" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="contentNativeFile" type="tns:capabilityType"
                    maxOccurs="1"/>
        <xs:element name="contentSharedNotes" type="tns:capabilityType"
                    maxOccurs="1"/>

  </xs:choice>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax"
                  minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

    </xs:sequence>

    <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
    <xs:attribute name="preferred" type="xs:boolean" use="optional"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>

  <xs:complexType name="serviceType">
    <xs:sequence>
      <xs:element name="capabilities" type="tns:capabilitiesType"
                  minOccurs="0" maxOccurs="1"/>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax"
                  minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>
```

```
      </xs:sequence>
      <xs:attribute name="uri" type="xs:anyURI"/>
      <xs:anyAttribute processContents="lax"/>
    </xs:complexType>

    <xs:complexType name="servicesType">
      <xs:sequence>
        <xs:element name="service" type="tns:serviceType"
                    minOccurs="1" maxOccurs="unbounded"/>

        <xs:sequence minOccurs="0" maxOccurs="1">
          <xs:sequence minOccurs="0" maxOccurs="unbounded">
            <xs:element ref="ct:delimiter"/>
            <xs:any namespace="##targetNamespace" processContents="lax"
                    minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:element ref="ct:end"/>
        </xs:sequence>
        <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

      </xs:sequence>
      <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
      <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
      <xs:anyAttribute processContents="lax"/>
    </xs:complexType>

    <xs:element name="services" type="tns:servicesType"/>

  </xs:schema>
```

7.6 userProperties Category

Location: http://schemas.microsoft.com/2006/09/sip/categories

```
    <?xml version="1.0" ?>
    <xs:schema
        targetNamespace="http://schemas.microsoft.com/2006/09/sip/categories"
        xmlns:tns="http://schemas.microsoft.com/2006/09/sip/categories"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified">

      <xs:import
          namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
          schemaLocation="commontypes.xsd"/>

      <xs:simpleType name="lineTypeEnum">
        <xs:restriction base="xs:token">
          <xs:enumeration value="Uc"/>
          <xs:enumeration value="Rcc"/>
          <xs:enumeration value="Dual"/>
          <xs:enumeration value="UcPrivate"/>
        </xs:restriction>
      </xs:simpleType>

      <xs:simpleType name="lineTypeEnumEx">
        <xs:union memberTypes="tns:lineTypeEnum xs:token" />
      </xs:simpleType>

      <xs:simpleType name="telephonyModeEnum">
        <xs:restriction base="xs:token">
          <xs:enumeration value="None"/>
          <xs:enumeration value="Uc"/>
          <xs:enumeration value="Rcc"/>
          <xs:enumeration value="Dual"/>
```

```
      <xs:enumeration value="NoAudioVideo"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="telephonyModeEnumEx">
    <xs:union memberTypes="tns:telephonyModeEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="phoneLineType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="lineType" type="tns:lineTypeEnumEx" use="required"/>
        <xs:attribute name="lineServer" type="xs:token"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="userPropertiesType">
    <xs:sequence>
      <xs:element name="lines" minOccurs="0" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="line" type="tns:phoneLineType"
                        minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="telephonyMode" type="tns:telephonyModeEnumEx"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="facsimileTelephoneNumber" type="xs:token"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="streetAddress" type="xs:token"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="l" type="xs:token" minOccurs="0" maxOccurs="1"/>
      <xs:element name="st" type="xs:token" minOccurs="0" maxOccurs="1"/>
      <xs:element name="countryCode" type="xs:token"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="postalCode" type="xs:token"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="wWWHomePage" type="xs:token" minOccurs="0" maxOccurs="1"/>
      <xs:element name="exumEnabled" type="xs:unsignedLong"
                  minOccurs="0" maxOccurs="1"/>
      <xs:element name="exumURL" type="xs:token" minOccurs="0" maxOccurs="1"/>
      <xs:element name="forwardingUrls" minOccurs="0" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:any processContents="lax" minOccurs="0"
                    maxOccurs="unbounded" namespace="##targetNamespace"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:element ref="ct:delimiter"/>
        <xs:element name="acpInformation" type="tns:acpProvisionType" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
    <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
```

```
      <xs:anyAttribute processContents="lax"/>
    </xs:complexType>

    <xs:element name="userProperties" type="tns:userPropertiesType"/>

    <xs:complexType name="acpProvisionType">
      <xs:sequence>
        <xs:element name="tollNumber" type="xs:token"/>
        <xs:element name="tollFreeNumber" type="xs:token" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="participantPassCode" type="xs:token"/>
        <xs:element name="domain" type="xs:token"/>
        <xs:element name="name" type="xs:token"/>
        <xs:element name="url" type="xs:anyURI" minOccurs="0"/>
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
  maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
        <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="default" type="xs:boolean" use="required"/>
      <xs:anyAttribute processContents="lax"/>
    </xs:complexType>

  </xs:schema>
```

7.7 legacyInterop Category

Location: http://schemas.microsoft.com/2006/09/sip/categories

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:tns="http://schemas.microsoft.com/2006/09/sip/categories"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:complexType name="legacyInteropType">
    <xs:attribute name="availability" type="xs:unsignedInt"/>
    <xs:attribute name="token" type="xs:token"/>
    <xs:attribute name="dndState" type="xs:boolean"/>
    <xs:anyAttribute processContents="lax"/>
  </xs:complexType>
  <xs:element name="legacyInterop" type="tns:legacyInteropType"/>
</xs:schema>
```

7.8 calendarData/workingHours Category

Location: http://schemas.microsoft.com/exchange/services/2006/types

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

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

```
      <xs:enumeration value="Sunday" />
      <xs:enumeration value="Monday" />
      <xs:enumeration value="Tuesday" />
      <xs:enumeration value="Wednesday" />
      <xs:enumeration value="Thursday" />
      <xs:enumeration value="Friday" />
      <xs:enumeration value="Saturday" />
      <xs:enumeration value="Day" />
      <xs:enumeration value="Weekday" />
      <xs:enumeration value="WeekendDay" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="DaysOfWeekType">
    <xs:list itemType="t:DayOfWeekType" />
  </xs:simpleType>
  <xs:complexType name="WorkingPeriod">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="DayOfWeek"
type="t:DaysOfWeekType" />
      <xs:element minOccurs="1" maxOccurs="1" name="StartTimeInMinutes"
type="xs:int" />
      <xs:element minOccurs="1" maxOccurs="1" name="EndTimeInMinutes" type="xs:int"
/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ArrayOfWorkingPeriod">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="WorkingPeriod"
type="t:WorkingPeriod" />
    </xs:sequence>
  </xs:complexType>
  <xs:annotation>
    <xs:documentation>
      This type is copied from exchange schema.
    </xs:documentation>
  </xs:annotation>
  <xs:complexType name="SerializableTimeZoneTime">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="Bias" type="xs:int" />
      <xs:element minOccurs="1" maxOccurs="1" name="Time" type="xs:string" />
      <xs:element minOccurs="1" maxOccurs="1" name="DayOrder" type="xs:short" />
      <xs:element minOccurs="1" maxOccurs="1" name="Month" type="xs:short" />
      <xs:element minOccurs="1" maxOccurs="1" name="DayOfWeek"
type="t:DayOfWeekType" />
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="SerializableTimeZone">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="Bias" type="xs:int" />
      <xs:element minOccurs="1" maxOccurs="1" name="StandardTime"
type="t:SerializableTimeZoneTime" />
      <xs:element minOccurs="1" maxOccurs="1" name="DaylightTime"
type="t:SerializableTimeZoneTime" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="WorkingHours">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="TimeZone"
type="t:SerializableTimeZone" />
```

```
      <xs:element minOccurs="1" maxOccurs="1" name="WorkingPeriodArray"
type="t:ArrayOfWorkingPeriod" />
    </xs:sequence>
  </xs:complexType>

  <xs:element name="WorkingHours" type="t:WorkingHours"/>

</xs:schema>
```

Location: http://schemas.microsoft.com/2006/09/sip/calendarData

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://schemas.microsoft.com/2006/09/sip/calendarData"
  xmlns:tns="http://schemas.microsoft.com/2006/09/sip/calendarData"
  xmlns:t="http://schemas.microsoft.com/exchange/services/2006/types"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ct="http://schemas.microsoft.com/2006/09/sip/commontypes"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <xs:import namespace="http://schemas.microsoft.com/exchange/services/2006/types"
             schemaLocation="calendardatatypes.xsd"/>

  <xs:import namespace="http://schemas.microsoft.com/2006/09/sip/commontypes"
             schemaLocation="commontypes.xsd"/>

  <xs:complexType name="calendarType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="1">
        <xs:element ref="t:WorkingHours"/>
        <xs:element name="freeBusy">
          <xs:complexType>
            <xs:simpleContent>
              <xs:extension base="xs:base64Binary">
                <xs:attribute name="startTime" type="xs:dateTime" use="required"/>
                <xs:attribute name="granularity" type="xs:duration" use="required"
/>
                <xs:attribute name="encodingVersion" type="xs:decimal"
use="required" />
                <xs:anyAttribute processContents="lax"/>
              </xs:extension>
            </xs:simpleContent>
          </xs:complexType>
        </xs:element>
      </xs:choice>

      <xs:sequence minOccurs="0" maxOccurs="1">
        <xs:sequence minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="ct:delimiter"/>
          <xs:any namespace="##targetNamespace" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:element ref="ct:end"/>
      </xs:sequence>
      <xs:element ref="ct:extension" minOccurs="0" maxOccurs="1"/>

    </xs:sequence>
    <xs:attribute name="mailboxID" type="xs:anyURI" use="required" />
```

```
      <xs:attribute name="majorVersion" type="xs:unsignedInt" use="optional" />
      <xs:attribute name="minorVersion" type="xs:unsignedInt" use="optional" />
      <xs:anyAttribute processContents="lax"/>
    </xs:complexType>
    <xs:element name="calendarData" type="tns:calendarType" />
  </xs:schema>
```

7.9 mwi Category

Location: http://schemas.microsoft.com/2006/09/sip/mwi

```
  <?xml version="1.0" encoding="UTF-8" ?>
  <xs:schema
      targetNamespace="http://schemas.microsoft.com/2006/09/sip/mwi"
      xmlns:tns="http://schemas.microsoft.com/2006/09/sip/mwi"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      elementFormDefault="qualified"
      attributeFormDefault="unqualified">

     <xs:complexType name="mwiType">
        <xs:sequence>
          <xs:any namespace="##other"
  processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="messageWaiting" type="xs:boolean" use="required"/>
        <xs:attribute name="unreadVoiceMailCount" type="xs:unsignedInt"
  use="optional"/>
        <xs:attribute name="readVoiceMailCount" type="xs:unsignedInt"
  use="optional"/>
        <xs:attribute name="majorVersion" type="xs:unsignedInt"
  use="optional" />
        <xs:attribute name="minorVersion" type="xs:unsignedInt"
  use="optional" />
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:complexType>
      <xs:element name="mwi" type="tns:mwiType"/>
  </xs:schema>
```

7.10 linkedPICContacts Category

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
   attributeFormDefault="unqualified"
   elementFormDefault="qualified"
   targetNamespace="http://schemas.microsoft.com/2006/09/sip/categories"
   xmlns:tns="http://schemas.microsoft.com/2006/09/sip/categories"
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="LinkedPicContactList">
   <xs:complexType>
    <xs:sequence>
     <xs:element minOccurs="0" maxOccurs="unbounded" name="LinkedPicContactEntry"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:complexType name="LinkedPicContactEntry">
   <xs:sequence>
    <xs:element
       name="secondary-identities"
       type="tns:secondary-identities"
       minOccurs="1" maxOccurs="1" nillable="false"/>
```

```
    <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="primary-Identity" type="xs:string" use="required"/>
   <xs:anyAttribute namespace="##any" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="secondary-identities">
   <xs:sequence>
    <xs:element name="secondary-identity" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:anyAttribute namespace="##any" processContents="lax"/>
  </xs:complexType>
</xs:schema>
```

8 Appendix C: Active Directory Search XML Schema

8.1 Active Directory Directory Search Request Schema

```
<?xml version="1.0" ?>
<xs:schema
    version="2.0" elementFormDefault="qualified"
    targetNamespace="http://schemas.microsoft.com/winrtc/2002/11/sip"
    xmlns:tns="http://schemas.microsoft.com/winrtc/2002/11/sip"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:ct="http://schemas.microsoft.com/sip/types">

  <xs:annotation>
    <xs:documentation>
      Server provides the ability to search for SIP users in an enterprise.
      This schema specifies the structure of the search request XML
      instance consumed by the server.

    </xs:documentation>
  </xs:annotation>

  <xs:import namespace="http://schemas.microsoft.com/sip/types"
             schemaLocation="common.xsd" />

 <xs:complexType name="searchRequest">
    <xs:sequence>
      <xs:element name="filter">
        <xs:complexType>
          <xs:attribute name="href" type="xs:IDREF">
            <xs:annotation>
              <xs:documentation>
                This value points to the element that contains the array of
                attributes to use for filtering search results returned
                from the Active Directory. The element that contains the
                results will contain an attribute named 'id' that has a
                value equal to this attribute
              </xs:documentation>
            </xs:annotation>
          </xs:attribute>
        </xs:complexType>
      </xs:element>

      <xs:element name="maxResults" type="xs:nonNegativeInteger">
        <xs:annotation>
          <xs:documentation>
            This value indicates the maximum number of search results
            requested by the protocol client. If this value is more than the maximum
            value configured by the administrator, then the maximum value
            set by the administrator is used. The "moreAvailable" element
            in the searchResponse will be set to true if the number of
            results returned by the Active Directory exceeds the number of
            results returned to the protocol client.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="searchFilters">
    <xs:annotation>
      <xs:documentation>
        The filters for a search operations can be any Active Directory
        attribute on a user object. The server takes all the filters
        specified and performs a logical 'AND' search using their values.
      </xs:documentation>
    </xs:annotation>
```

```
    <xs:sequence>
      <xs:element name="row" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="attrib" type="tns:searchAttribute" use="required" />
          <xs:attribute name="value" type="xs:string" use="required" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>

    <xs:attribute name="id" type="xs:ID" use="required">
      <xs:annotation>
        <xs:documentation>
          The value of this attribute equals the value specified in the
          'href' attribute of the filter element in the searchRequest.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>

  <xs:element name="directorySearch" type="tns:searchRequest" />
  <xs:element name="Array" type="tns:searchFilters" />
</xs:schema>
```

8.1.1 Directory Search SOAP Envelope schema

```
<xs:schema xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope" attributeFormDefault="unqualified"
elementFormDefault="qualified" targetNamespace="http://schemas.xmlsoap.org/soap/envelope"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:import namespace="http://schemas.microsoft.com/winrtc/2002/11/sip"
schemaLocation="Directorysearchrequest.xsd"/>
  <xs:element name="Envelope">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Body">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="m:directorySearch" />
              <xs:element ref="m:Array" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

8.2 Active Directory Directory Search Response Schema

```
<?xml version="1.0" ?>
<xs:schema
    version="2.0" elementFormDefault="qualified"
    targetNamespace="http://schemas.microsoft.com/winrtc/2002/11/sip"
    xmlns:tns="http://schemas.microsoft.com/winrtc/2002/11/sip"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:ct="http://schemas.microsoft.com/sip/types">

  <xs:annotation>
    <xs:documentation>
      Server provides the ability to search for SIP users in an enterprise.
      This schema specifies the structure of the search results XML
      instance returned by the server.
    </xs:documentation>
  </xs:annotation>
```

```
  <xs:import namespace="http://schemas.microsoft.com/sip/types" schemaLocation="common.xsd"
/>

  <xs:complexType name="searchResponse">
    <xs:sequence>
      <xs:element name="moreAvailable" type="xs:boolean">
        <xs:annotation>
          <xs:documentation>
            This value indicates whether more results are available.
        </xs:documentation>
        </xs:annotation>
      </xs:element>

      <xs:element name="returned" type="xs:nonNegativeInteger">
        <xs:annotation>
          <xs:documentation>
            This value indicates how many results are returned
          </xs:documentation>
        </xs:annotation>
      </xs:element>

      <xs:element name="value">
        <xs:complexType>
          <xs:attribute name="href" type="xs:IDREF" use="required">
            <xs:annotation>
              <xs:documentation>
                This value points to the element that contains the results.
                The element that contains the results will contain an
                attribute named 'id' that has a value equal to this
                attribute
              </xs:documentation>
            </xs:annotation>
          </xs:attribute>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="searchResults">
    <xs:sequence>
      <xs:element name="row" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="uri" type="ct:sipURI" use="required" />
          <xs:attribute name="displayName" type="ct:displayName" use="required" />
          <xs:attribute name="title" type="ct:title" use="required" />
          <xs:attribute name="office" type="ct:office" use="required" />
          <xs:attribute name="phone" type="ct:phone" use="required" />
          <xs:attribute name="company" type="ct:company" use="required" />
          <xs:attribute name="city" type="ct:city" use="required" />
          <xs:attribute name="state" type="ct:state" use="required" />
          <xs:attribute name="country" type="ct:country" use="required" />
          <xs:attribute name="email" type="ct:email" use="required" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>

    <xs:attribute name="id" type="xs:ID" use="required" >
      <xs:annotation>
        <xs:documentation>
          The value of this attribute equals the value specified
          in the 'href' attribute of the filter element in the
          searchResponse.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

  </xs:complexType>
```

```
    <xs:complexType name="searchRequest">
      <xs:sequence>
        <xs:element name="filter">
          <xs:complexType>
            <xs:attribute name="href" type="xs:IDREF">
              <xs:annotation>
                <xs:documentation>
                  This value points to the element that contains the array of
                  attributes to use for filtering search results returned
                  from the Active Directory. The element that contains the
                  results will contain an attribute named 'id' that has a
                  value equal to this attribute
                </xs:documentation>
              </xs:annotation>
            </xs:attribute>
          </xs:complexType>
        </xs:element>

        <xs:element name="maxResults" type="xs:nonNegativeInteger">
          <xs:annotation>
            <xs:documentation>
              This value indicates the maximum number of search results
              requested by the protocol client. If this value is more than the maximum
              value configured by the administrator, then the maximum value
              set by the administrator is used. The "moreAvailable" element
              in the searchResponse will be set to true if the number of
              results returned by the Active Directory exceeds the number of
              results returned to the protocol client.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="searchFilters">
      <xs:annotation>
        <xs:documentation>
          The filters for a search operations can be any Active Directory
          attribute on a user object. The server takes all the filters
          specified and performs a logical 'AND' search using their values.
        </xs:documentation>
      </xs:annotation>

      <xs:sequence>
        <xs:element name="row" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:attribute name="attrib" type="tns:searchAttribute" use="required" />
            <xs:attribute name="value" type="xs:string" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>

      <xs:attribute name="id" type="xs:ID" use="required">
        <xs:annotation>
          <xs:documentation>
            The value of this attribute equals the value specified in the
            'href' attribute of the filter element in the searchRequest.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>


    <xs:element name="directorySearch" type="tns:searchResponse" />
    <xs:element name="Array" type="tns:searchFilters" />
  </xs:schema>
```

8.2.1 Directory Search SOAP Envelope response schema

```
<xs:schema xmlns:m="http://schemas.microsoft.com/winrtc/2002/11/sip" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" attributeFormDefault="unqualified"
elementFormDefault="qualified" targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:import namespace="http://schemas.microsoft.com/winrtc/2002/11/sip" />
  <xs:element name="Envelope">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Body">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="m:directorySearch" />
              <xs:element ref="m:Array" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

8.3 COMMON.XSD

```
<?xml version="1.0" ?>
<xs:schema id="contact" version="2.0"
           targetNamespace="http://schemas.microsoft.com/sip/types"
           xmlns:tns="http://schemas.microsoft.com/sip/types"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:simpleType name="adAttribute">
    <xs:restriction base="xs:token">
      <xs:minLength value="1" />
      <xs:pattern value="\w+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="sipURI">
    <xs:annotation>
      <xs:documentation>
        The format of a SIP URI is sip:user@host. The user portion
        of the URI is treated as case-sensitive while the host portion
        is treated as case-insensitive.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:anyURI">
      <xs:maxLength value="454" />
      <xs:pattern value="sip:\w+@(\w+.)*\w+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="displayName">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'displayName' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="email">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
```

```
        'mail' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="phone">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'telephoneNumber' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="title">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'title' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="office">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'physicalDeliveryOfficeName' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="company">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'company' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="city">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'l' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="state">
    <xs:annotation>
      <xs:documentation>
        This value is retrieved by the server from the Active Directory
        'st' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="country">
    <xs:annotation>
      <xs:documentation>
```

```
        This value is retrieved by the server from the Active Directory
        'c' attribute on the user object.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="tns:adAttribute" />
  </xs:simpleType>

  <xs:simpleType name="searchMacro">
    <xs:restriction base="xs:token">
      <xs:enumeration value="msRTCURI">
        <xs:annotation>
          <xs:documentation>
            When specified, the server will create a filter expression of
            the form (|(msRTCSIP-PrimaryUserAddress=&lt;value&gt; or use
            CDATA signal*)(msRTCSIP-SecondaryUserAddress=&lt;value&gt;
            or use CDATA signal*))
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="msRTCMail">
        <xs:annotation>
          <xs:documentation>
            When specified, the server will create a filter expression
            of the form (|(mail=&lt;value&gt; or use CDATA signal
            *)(proxyAddresses=smtp: &lt;value&gt; or use CDATA signal
            *))
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="searchAttribute">
    <xs:union memberTypes="tns:adAttribute tns:searchMacro" />
  </xs:simpleType>

</xs:schema>
```

9 Appendix D: MSRTC XML Schemas

See [MS-SIP] section 6 for the full schema of "text/xml+msrtc.pidf" presence document format.

The **aggregate** element is the last element in the **aggregatedPresenceDoc** type described in [MS-SIP] section 6.

```
<xs:complexType name="aggregatedPresenceDoc">
…….
…..

<xs:element name="userInfo" …
<xs:element name="devices" …….
<xs:element name="aggregate" type="tns:aggregateType"/>
</xs:sequence>
<xs:attribute name="uri" type="ct:sipURI" use="required"/>
</xs:complexType>


This section contains the schema for the aggregate element extension


<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  targetNamespace="http://schemas.microsoft.com/2002/09/sip/presence"
  xmlns:tns="http://schemas.microsoft.com/2002/09/sip/presence"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <xs:simpleType name="activityTokenTypeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="online"/>
      <xs:enumeration value="busy"/>
      <xs:enumeration value="do-not-disturb"/>
      <xs:enumeration value="be-right-back"/>
      <xs:enumeration value="away"/>
      <xs:enumeration value="offline"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="activityTokenTypeEnumEx">
    <xs:union memberTypes="tns:activityTokenTypeEnum xs:token" />
  </xs:simpleType>

  <xs:complexType name="stateType" abstract="true">
    <xs:simpleContent>
      <xs:extension base="tns:activityTokenTypeEnumEx">
        <xs:attribute name="avail" type="xs:unsignedInt"/>
        <xs:anyAttribute processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:complexType name="userState">
    <xs:complexContent>
      <xs:extension base="tns:stateType"/>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="statesType">
    <xs:sequence>
      <xs:element name="state" type="tns:stateType"
                  minOccurs="1" maxOccurs="1"/>
      </xs:sequence>
  </xs:complexType>
```

```
    <xs:complexType name="aggregateType">
      <xs:sequence>
        <xs:element name="states" type="tns:statesType"
                    minOccurs="1" maxOccurs="1"/>
      </xs:sequence>
    </xs:complexType>

    <xs:element name="aggregate" type="tns:aggregateType"/>

  </xs:schema>
```

10 Appendix E: Contact Management XML Schemas

See [MS-SIP] section 9 for the full contact management schema.

The **deltaNum** attribute is the last attribute in the **fullContactList** type described in [MS-SIP] section 9.

```
<xs:complexType name="fullContactList">
…….
…..

    <xs:element name="contact" type="tns:contact"
                minOccurs="0" maxOccurs="unbounded">
       <xs:annotation>
         <xs:documentation>
           Although the schema allows for an unbounded number of contacts, the
    administrator can configure a server to disallow more than a certain number of
contacts.
         </xs:documentation>
       </xs:annotation>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="deltaNum" type="tns:contactDeltaNm" use="required"/>
</xs:complexType>

<xs:element name="contactList" type="tns:fullContactList" />
```

This section contains the schema for the **fullContactList** type extension<50>

```
<xs:complexType name="fullContactList">
  <xs:sequence>
    <xs:element name="group" type="tns:group"
                minOccurs="0" maxOccurs="64" />
    <xs:element name="contact" type="tns:contact"
                minOccurs="0" maxOccurs="unbounded" >
      <xs:annotation>
        <xs:documentation>
          Although the schema allows for an unbounded number of
          contacts, the administrator can configure a server to
          disallow more than a certain number of contacts.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="deltaNum" type="tns:contactDeltaNum" use="required" />
  <xs:attribute name="ucsMode" type="tns:contactUcsMode" use="required" />

</xs:complexType>

<xs:element name="contactList" type="tns:fullContactList" />
```

The **fullContactList** type has been extended with a new attribute called ucsMode as shown above.

The **prevDeltaNum** attribute is the last attribute in the **deltaContactList** type described in [MS-SIP] section 9.

```
<xs:complexType name="deltaContactList">
…….
…..


  <xs:attribute name="deltaNum" type="tns:contactDeltaNum" use="required" >
    <xs:annotation>
      <xs:documentation>
        The value of this attribute is the new delta number
        after the SERVICE operation was performed.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="prevDeltaNum" type="tns:contactDeltaNum" use="required" >
    <xs:annotation>
      <xs:documentation>
        The value of this attribute equals the value of the
        delta number specified in the SERVICE operation.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>

<xs:element name="contactDelta" type="tns:deltaContactList" />
```

This section contains the schema for the **deltaContactList** type extension<51>

```
<xs:complexType name="deltaContactList">
  <xs:choice>
    <xs:element name="addedGroup" type="tns:group" />
    <xs:element name="modifiedGroup" type="tns:group" />
    <xs:element name="addedContact" type="tns:contact" />
    <xs:element name="modifiedContact" type="tns:contact" />
    <xs:element name="deletedGroup">
      <xs:complexType>
        <xs:attribute name="id" type="tns:groupID" use="required" />
      </xs:complexType>
    </xs:element>
    <xs:element name="deletedContact">
      <xs:complexType>
        <xs:attribute name="uri" type="tns:sipURI" use="required" />
      </xs:complexType>
    </xs:element>
  </xs:choice>
  <xs:attribute name="deltaNum" type="tns:contactDeltaNum" use="required" >
    <xs:annotation>
      <xs:documentation>
        The value of this attribute is the new delta number
        after the SERVICE operation was performed.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="prevDeltaNum" type="tns:contactDeltaNum" use="required" >
    <xs:annotation>
      <xs:documentation>
        The value of this attribute equals the value of the
        delta number specified in the SERVICE operation.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="ucsMode" type="tns:contactUcsMode" use="required" />
```

```
  </xs:complexType>

  <xs:element name="contactDelta" type="tns:deltaContactList" />
```

The **deltaContactList** type has been extended with a new attribute called ucsMode as shown above.

The definition for new type contactUcsMode is shown below

```
<xs:simpleType name="contactUcsMode">
  <xs:restriction base="xs:string">
    <xs:annotation>
      <xs:documentation>
        The server maintains the ucs mode for every user. The ucs mode indicates
        if the contact list for the user is stored on the server or another
        external store such as Exchange. The ucs mode can be disabled, allowed,
        migrating or migrated. If the mode is disabled, the contact list is
        stored on the server. If the mode is allowed, it indicates that the
        contact list is stored on the server but will be moved to the external
        store anytime. If the ucs mode is set to migrating, it indicates that the
        contact list is in the process of being migrated to the external store.
        If the mode is migrated, migration has already happened and contact list
        is now on the external store. When the ucs mode is set to migrating or
        migrated, all operations that will modify the contact list are rejected
        by the server though an implementation may choose to suppose
        modifications while the contact list is migrating and/or after it has
        migrated.
      </xs:documentation>
    </xs:annotation>
  </xs:restriction>
</xs:simpleType>
```

11 Appendix F: Product Behavior

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

- Microsoft Office Communications Server 2007
- Microsoft Office Communicator 2007
- Microsoft Office Communications Server 2007 R2
- Microsoft Office Communicator 2007 R2
- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Lync Server 2013
- Microsoft Skype for Business (formerly Lync 2013)
- 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 2.2.1: Supported in Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2 only.

<2> Section 2.2.2.4.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<3> Section 2.2.2.7.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This element is not supported.

<4> Section 2.2.2.7.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<5> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<6> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This attribute is not supported.

<7> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This attribute is not supported.

<8> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This attribute is not supported.

<9> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This attribute is not supported.

<10> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007: This attribute is not supported.

<11> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This element is not supported.

<12> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This element is not supported.

<13> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This element is not supported.

<14> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This element is not supported.

<15> Section 2.2.2.7.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: This element is not supported.

<16> Section 2.2.2.7.4: Products other than Office Communications Server 2007, Office Communicator 2007: This behavior is updated to support features as described by the MSDN Knowledgebase Article #967673, "Description of the Communicator 2007 R2 hotfix rollup package: April 2009". The hotfix rollup package added support for the "NoAudioVideo" token.

<17> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007: This value is not supported.

<18> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007: This value is not supported.

<19> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007: This value is not supported.

<20> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This value is not supported.

<21> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This value is not supported.

<22> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This value is not supported.

<23> Section 2.2.2.7.5: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This value is not supported.

<24> Section 2.2.2.7.8: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<25> Section 2.2.2.7.9: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<26> Section 2.2.2.7.10: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<27> Section 2.2.2.7.11: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<28> Section 2.2.2.7.12: Office Communications Server 2007, Office Communications Server 2007 R2, Lync Server 2010, Lync Server 2013: This behavior is not supported.

<29> Section 2.2.2.7.13: This behavior is supported only by Skype for Business (formerly Lync 2013).

<30> Section 3.1: Supported in Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2 only.

<31> Section 3.2.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: The registered category names are "note", "state", "contactCard", "device", "services", "userProperties", "routing", and "legacyInterop". The agreed-upon schema for each of these categories are listed in section 7.

<32> Section 3.2.2: For Office Communications Server 2007, the default period of this timer is five minutes. 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.

<33> Section 3.2.2: For Office Communications Server 2007, the default period of this timer is one minute. 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.

<34> Section 3.2.5.7: This behavior is supported only by Lync Server 2013.

<35> Section 3.3.5.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<36> Section 3.3.5.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<37> Section 3.5.5.5: Lync Server 2010 does not return a 403 Forbidden response in this case, but will return the same body as the request to the client.

<38> Section 3.6.5.3: Lync Server 2010 does not return a 403 Forbidden response in this case, but will return the same body as the request to the client.

<39> Section 3.7.5.5: If this value is not obtained successfully, the server sends a 500 error response.

<40> Section 3.7.5.5: If this value is not obtained successfully, the server sends a 500 error response. In Lync Server 2010 and Lync 2010, this element is empty in the response notification.

<41> Section 3.7.5.5: If this value is not obtained successfully, the server sends a 500 error response. In Lync Server 2010 and Lync 2010, this element is empty in the response notification.

<42> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<43> Section 3.8.5.1.2.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<44> Section 3.8.5.1.2.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<45> Section 3.8.5.1.2.6: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<46> Section 3.8.5.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<47> Section 3.8.5.4: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<48> Section 3.9.5.4: Lync Server 2010 does not return a 403 Forbidden response in this case, but will instead return the same body as the request to the client.

<49> Section 3.10: This behavior is supported only by Lync Server 2013.

<50> Section 10: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010: This behavior is not supported.

<51> Section 10: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010: This behavior is not supported.

12 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
2.2.2.7.5 contactCard	Updated the description of displayADPhoto.	N	Content update.
2.2.2.7.6 legacyInterop	Refined the description for dndState.	N	Content update.
2.2.2.7.13 Persistent chat categories	Removed the word "certain".	N	Content update.
3.8 Enhanced Presence Aggregation Details	Refined the description which contains "certain".	N	Content update.
11 Appendix F: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

13 Index

A

C

D

E

F

G

H

I

L

M

T

U

V

X