[MS-ECREST]:

Unified Communications Event Channel Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
3/30/2015	1.0	New	Released new document.
9/4/2015	1.0	None	No changes to the meaning, language, or formatting of the technical content.
7/15/2016	1.1	Minor	Clarified the meaning of the technical content.
9/14/2016	1.1	None	No changes to the meaning, language, or formatting of the technical content.
12/15/2016	2.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Unified Communications Event Channel Protocol describes a mechanism which web applications can use to retrieve notifications about changes to resources.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

byte order mark: A Unicode character that is used to indicate that text is encoded in UTF-8, UTF-16, or UTF-32.

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

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

Representational State Transfer (REST): A software architecture implementation for distributed hypermedia systems, such as the World Wide Web.

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].

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

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

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

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

1.2 References

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

1.2.1 Normative References

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

[MS-OCDISCWS] Microsoft Corporation, "Lync Autodiscover Web Service Protocol".

[MS-OCSMP] Microsoft Corporation, "Microsoft Online Conference Scheduling and Management Protocol".

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

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

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

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

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

1.2.2 Informative References

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

1.3 Overview

This protocol is used to receive events and other changes to resources that can occur even without any specific action taken by the web application. A typical scenario for this protocol is to detect new invitations to conversations, participants coming and going from conversations, and media being added or removed from a conversation.

1.4 Relationship to Other Protocols

This protocol transmits request and response messages by using **HTTP**, as described in [RFC2616], or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [RFC2818].

Access to the Event Channel Web Service is discovered through the Lync Autodiscover Web Service. The discovery service protocol is described in the [MS-OCDISCWS] document.

This protocol is accessible to authenticated users, either directly via a client application or indirectly via a trusted server application. The authentication service protocols are described in the [MS-OCAUTHWS] document.

The following diagram shows the underlying messaging and transport stack used by the protocol:

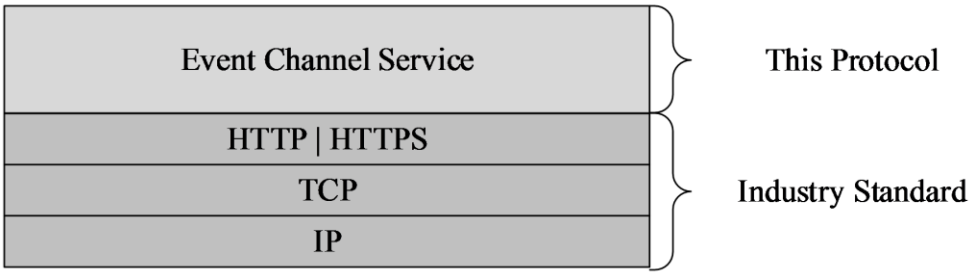

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a protocol server that is configured to listen for **HTTP** or **HTTPS** requests and a protocol client that knows the Request-URI of the protocol server.

1.6 Applicability Statement

This protocol is applicable for the following scenarios:

- Using a UCWA API that has an asynchronous or non-blocking operation
- Working with conversations, participants, and media types within conversations
- Working with presence or other protocols that can produce events not caused by any specific action on behalf of the application.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the area of supported transports. This protocol can use **HTTP** or **HTTPS** as a transport. For more details, see section 2.1.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

This protocol is based on **Representational State Transfer (REST)** principals. As such, messages are transported using **HTTP**, as specified in [RFC2616], or **HTTPS**, as specified in [RFC2818]. The service SHOULD be served on ports 80 and 443 respectively, but MAY be served on other ports. For specific port information, please contact your service provider.

Protocol messages are text-based and MUST be UTF-8 encoded. Messages MUST NOT contain a **byte order mark**. The byte order mark is a Unicode character used to signal the byte order of a text file or stream.

2.2 Common Data Types

This section contains common definitions used by this protocol. The syntax of the definitions uses **XML schema** as defined in section 6.

2.2.1 Namespaces

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

Prefix	Namespace URI	Reference
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
tns	http://schemas.microsoft.com/rtc/2012/03/ucwa	

2.2.2 Common URI Parameters

The following table summarizes the set of common **URI** parameters defined by this specification.

URI Parameter	Description
ack	This parameter is added automatically to all URIs returned by the events service either by a GET on the application resource, or as the **next** link in an event response. The parameter and its value SHOULD be treated as opaque and SHOULD NOT be added or altered by the application.
timeout	The time in seconds that the HTTP GET request SHOULD block before timing out if no events are received. For desktop applications, a larger value such as 900 seconds MAY be used.
medium	The aggregation interval for medium priority events in seconds. The default value is 5 seconds and is sufficient for most applications. Maximum value is 30 * 60 seconds.
low	The aggregation interval for low priority events in seconds. The default value is 15 seconds and is sufficient for most applications. Maximum value is 30 * 60 seconds.

URI Parameter	Description
priority	The priority of the request relative to other pending GET requests (P-GETS) with the same Ack ID. A higher priority requests replaces a lower priority request.

2.2.3 Elements

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

Element	Description
events	The top-level element that contains the event information returned on the event channel.

2.2.3.1 events

The events element is the top-level element that contains the event information returned on the event channel.

```
<xs:element name="events" type="tns:EventsType" />
```

2.2.4 Complex Types

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

Complex Type	Description
EventsType	Contains event information being returned to the application.
EventType	A resource has been added, updated, deleted, started, or completed.
InType	Contains information on referencing a resource if it was added or removed from a collection.
SenderType	Specifies the event that was sent by the sender.

2.2.4.1 EventsType

The **EventsType** complex type is the type of top-level element that contains the event information returned on the event channel.

```
<xs:complexType name="EventsType">
  <xs:sequence>
    <xs:element name="link" type="tns:LinkType" minOccurs="1" maxOccurs="1" />
```

```
      <xs:element maxOccurs="unbounded" name="sender" type="tns:SenderType" />
    </xs:sequence>
    <xs:attribute name="href" type="xs:string" use="required" />
  </xs:complexType>
```

Element	Type	Description
link	tns:LinkType ([MS-OCSMP] section 2.2.4.5)	A reference to a related event.
sender	tns:SenderType (section 2.2.4.4)	Event that was sent by the sender.

Attribute	Type	Description
hrel	xs:string ([XMLSCHEMA2])	The URI of the event itself.

2.2.4.2 EventType

The **EventType** complex type is included when a resource has been added, updated, deleted or an operation has started or completed.

```
<xs:complexType name="EventType">
    <xs:sequence minOccurs="0">
      <xs:element minOccurs="0" name="in" type="tns:InType" />
      <xs:element minOccurs="0" name="resource" type="tns:ResourceType" />
      <xs:element minOccurs="0" name="reason" type="tns:ErrorType" />
    </xs:sequence>
    <xs:attribute name="rel" type="xs:string" use="required" />
    <xs:attribute name="href" type="xs:string" use="required" />
    <xs:attribute name="title" type="xs:string" use="optional" />
    <xs:attribute name="context" type="xs:string" use="optional" />
  </xs:complexType>
```

Element	Type	Description
in	tns:InType (section 2.2.4.3)	An optional element if the resource was added or removed from a collection.
resource	tns:ResourceType ([MS-OCSMP] section 2.2.4.7)	The resource information.
reason	tns:ErrorType ([MS-OCSMP] section 2.2.4.4)	Describes additional information about the failure.

Attribute	Type	Description
rel	xs:string ([XMLSCHEMA2])	A value that provides the type of the resource or operation.
hrel	xs:string ([XMLSCHEMA2])	The URI of the resource or operation.
title	xs:string ([XMLSCHEMA2])	An optional title of the resource or operation.
context	xs:string ([XMLSCHEMA2])	An optional context of the resource or operation.

2.2.4.3 InType

The **InType** complex type is included if the resource was added or removed from a collection.

```
<xs:complexType name="InType" >
  <xs:attribute name="rel" type="xs:string" use="required" />
  <xs:attribute name="href" type="xs:string" use="required" />
  <xs:attribute name="title" type="xs:string" use="required" />
</xs:complexType>
```

Attribute	Type	Description
rel	xs:string ([XMLSCHEMA2])	A value that provides the type of the collection.
hrel	xs:string ([XMLSCHEMA2])	The URI of the collection.
title	xs:string ([XMLSCHEMA2])	An optional title of the collection.

2.2.4.4 SenderType

The **SenderType** complex type specifies the event that was sent by the sender.

```
<xs:complexType name="SenderType">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element name="added" type="tns:EventType" />
      <xs:element name="updated" type="tns:EventType" />
      <xs:element name="deleted" type="tns:EventType" />
      <xs:element name="started" type="tns:EventType" />
      <xs:element name="completed" type="tns:EventType" />
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="rel" type="xs:string" use="required" />
  <xs:attribute name="href" type="xs:string" use="required" />
</xs:complexType>
```

Element	Type	Description
added	tns:EventType (section 2.2.4.2)	A resource has been added.
updated	tns:EventType (section 2.2.4.2)	A resource has been updated.
deleted	tns:EventType (section 2.2.4.2)	A resource has been deleted.
started	tns:EventType (section 2.2.4.2)	An operation has started.
completed	tns:EventType (section 2.2.4.2)	An operation has completed。

Attribute	Type	Description
rel	xs:string ([XMLSCHEMA2])	A value that provides the type of the event target.

Attribute	Type	Description
hrel	xs:string ([XMLSCHEMA2])	The URI of the event target.

2.2.5 Attributes

This specification uses the "rel" and "href" attributes as specified in [MS-OCSMP] section 2.2.6.

2.2.6 Common Data Structures

This specification uses the **ErrorType** data structure as specified in [MS-OCSMP] section 2.2.4.4.

3 Protocol Details

3.1 Server Details

The basic unit for operations in the Event Channel protocol is an Event. Applications will be able to discover changes to resources as well as the properties of a resource thus allowing applications to create a UI that is dynamically adjustable.

The entirety of the Event Channel will be accessible via the **HTTP** protocol. This protocol is exposed via the Unified Communications Web API (UCWA) web component of the protocol server.

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 help explain how the protocol behaves. This specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this specification.

Web developers can use the Event Channel protocol to create web pages that react to changes to the communication and collaboration features of a protocol server.

3.1.1.1 Introduction

An Event Channel Protocol client application first queries the autodiscovery service, as specified in [MS-OCDISCWS], to find the home server for the user that is associated with the application.

After the home server is located, an Event Channel client application interacts with the EventChannel server, using the **HTTP** protocol.

At the most basic level, an Event Channel client application communicates with the Event Channel server by sending HTTP requests (GET) to the service, which sends back HTTP responses. Each HTTP request that is sent to the Event Channel Protocol server includes the URL of a specific resource. The response includes links to related resources, in the form of URLs.

3.1.1.2 Basic Concepts

The basic concepts specified in [MS-OCSMP] section 3.1.1.2 apply to this protocol.

3.1.2 Timers

None.

3.1.3 Initialization

This protocol requires creating an application resource as a prerequisite to get the initial resource URI.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The following table lists all the resources exposed by the Event Channel protocol service.

Resource	Description
application	Represents one instance of an application that is run by a user on a specific device.
applications	Represents a factory in which individual application resources are created.
events	Represents a collection of event data for processing by the application.

3.1.5.1 application

The application resource is specified in [MS-OCSMP] section 3.1.5.1.

3.1.5.2 applications

The applications resource is specified in [MS-OCSMP] section 3.1.5.2.

3.1.5.3 events

The events resource supports only the HTTP GET method to retrieve events waiting for the application.

Token	Media types	HTTP method
events	application/xml application/json	GET

Unlike other HTTP GET operations, a timeout can be specified and the GET will block until events are available or the timeout period is reached. For this reason, the event channel needs its own separate HTTP channel or queue because it needs to run independently of any other HTTP requests to change state at the server.

There SHOULD be at most one active GET request on the event channel at any time.

3.1.5.3.1 Request Body

Request header	Usage	Value
Accept	Response content-type negotiation.	application/xml application/json

The request body SHOULD be empty.

3.1.5.3.2 Response Body

Response header	Usage	Value
Content-type	Response content-type.	application/xml application/json

The response body will contain an **events** data structure (section 2.2.3.1).

The response to this operation can result in the following status codes.

Status code	Description
400	Query parameters invalid or out of range.
401	Unauthorized
403	Forbidden
404	Subcode = ApplicationNotFound. The server application does not exist at all. The application SHOULD recreate the application using a POST request on the applications if it plans to continue running.
409	Subcode = PGet replaced. The client application submitted another HTTP GET with the same URI causing the current one to be released. This might be an indication another instance of the application might be running if the application did not replace the request intentionally.
410	The requested resource is no longer available at the server and no forwarding address is known.
500	There is a general service failure at the http proxy or server.
502	There is an error routing the request or a dependent service is down.
503	Can be returned from the web proxy if the service is in maintenance mode.

3.1.5.3.3 Getting the next link

Upon receiving a successful response to the GET request, the application MUST extract the link given with **rel** attribute value equal to "next" for its next request to retrieve events. The **URI** is to be treated as opaque and the ack URI parameter SHOULD NOT be interpreted or altered. However, additional query parameters can be appended.

3.1.5.3.4 Basics of event processing

Events are grouped by the resource responsible for sending them. For example, the communication resource is responsible for sending updates for conversations, and conversations are responsible for sending conversation related events such as participants joining or leaving the conversation.

Events are guaranteed to arrive in an order that makes sense for processing organized by the same sender. You will not get a deleted event for a resource before you get the added event, and you won't get updated events for resources that haven't been added. Resources that contain others will be added first before their inner resources are added.

When receiving deleted events for resources that contain other resources, the application SHOULD assume that the contained resources are also deleted. As an example, a deleted conversation means the participants are also deleted. You might not actually receive a delete event for each participant.

For each event type, you might receive a current snapshot of the resource that was updated embedded in the event information for common scenarios. This saves a network round trip to get the resource data by using an HTTP GET.

If you don't get the resource inline, you will get a link to the resource described by the event. The **rel** attribute can be used to find the resource as a link or an enclosed resource. Applications SHOULD be prepared for both cases.

For resources that are not always available, they will go through the sequence of being added, updated then deleted either explicitly or by its parent resource. Operations that are running are generally are started, optionally updated, then at some point they complete either with success or failure. The events you get returned to the application are affected by aggregation algorithms at the server so you might not always see every event depending on the timing.

3.1.5.3.5 Event Aggregation

Events are collected in a queue on the server as they arrive. However, to optimize processing and network traffic the server might consolidate two or events together in a process called aggregation. Events are generally aggregated when the net processing of the results of the single notification would be the same as the sequence that was aggregated.

Examples:

1. An added event followed by an updated event could be consolidated to a single added event with the current information about the resource.

2. Multiple updated events could be consolidated into one updated event with the most current information.

3. Started event followed by completed event happens on success, where on operation failure, only the completed event might be reported.

The aggregation behavior might lead to unexpected behaviors, such as apparently missing intermediate states of resources. You might see a resource transition from a "Disconnected" state to "Connected" without seeing "Connecting".

Similarly code that looks for a link only in an updated event might not work if it appears in the added event due to aggregation.

3.1.5.3.6 Posting another Event Channel request

After processing, the data in the response to the GET request the application SHOULD post another GET to get the next batch of events using the next link from the most recent response potentially adding optional parameters other than ack to the URI as needed. The most common parameter to add to the **URI** is the timeout parameter.

3.1.5.3.7 Detecting query synchronization issues

If the server application is active and the client sends a request using a **URI** that is out of order (lower than the earliest one that the server has, or higher than the latest one), the response will consist of a single link with rel="resync". Following the link will give the client the first unacknowledged event set.

Such a response usually indicates the presence of a client application bug, which SHOULD NOT occur in normal operation.

The client application SHOULD be prepared to see the resync link in any response. If the client application receives such a link, the correct action for it is to clear the caches for all transient data (including active conversations and subscriptions) and refetch the needed data.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

4.1 Creating application

Before using the Event Channel Protocol, it is necessary to create an application resource and get the events resource **URI** in the response.

4.1.1 HTTP Request

```
POST https://ext.vdomain.com:4443/ucwa/oauth/v1/applications HTTP/1.1
User-Agent: UcwaClient/1.0
Authorization: Bearer cwt=AAEBHAEFAAAAAAAFFQAAAPmQRuRgha2wB2cEg...
Accept: application/xml
Content-Type: application/xml
Host: ext.vdomain.com:4443
Content-Length: 316
Expect: 100-continue
<?xml version="1.0" encoding="utf-8"?>
<input xmlns="http://schemas.microsoft.com/rtc/2012/03/ucwa">
<property name="culture">en-US</property>
<property name="endpointId">e80dc357-19bb-418d-93bf-1ecb5135d43f</property>
<property name="userAgent">UcwaClient/1.0</property>
<property name="type">Phone</property>
</input>
```

4.1.2 HTTP Response

```
HTTP/1.1 201 Created
Cache-Control: private
Via: 1.1 accessproxy.vdomain.com RtcExt
Content-Length: 5063
Content-Type: application/xml; charset=utf-8
Expires: Thu, 22 Jan 2015 06:43:24 GMT
ETag: "766942815"
X-MS-Server-Fqdn: SERVER.vdomain.com
X-MS-Correlation-Id: 2147483652
client-request-id: 13efd9fe-d041-4729-af67-9f97abb81a18
X-MS-Correlation-Id: 2147483679
client-request-id: bfcaa11b-6cb2-44f2-9f56-6569e15c643b
Strict-Transport-Security: max-age=31536000; includeSubDomains
Date: Wed, 21 Jan 2015 22:43:23 GMT
<?xml version="1.0" encoding="utf-8"?>
<resource rel="application" href="/ucwa/oauth/v1/applications/211177894191"
xmlns="http://schemas.microsoft.com/rtc/2012/03/ucwa">
    <link rel="policies" href="/ucwa/oauth/v1/applications/211177894191/policies"
/><link rel="batch" href="/ucwa/oauth/v1/applications/211177894191/batch" />
    <link rel="events" href="/ucwa/oauth/v1/applications/211177894191/events?ack=1" />
    <property name="culture">en-US</property>
    <property name="userAgent">UcwaClient/1.0</property>
    <property name="type">Phone</property><property name="etag">766942815</property>
<!-- Other content removed for clarity -- >
</resource>
```

4.2 Getting event data

In this example, a protocol client sends a request to the protocol server to receive notifications, such as the status of some background operations it requested earlier.

4.2.1 HTTP Request

```
GET
https://ext.vdomain.com:4443/ucwa/oauth/v1/applications/211177894191/events?ack=1&time
out=900 HTTP/1.1
User-Agent: UcwaClient/1.0
Authorization: Bearer cwt=AAEBHAEFAAAAAAAFFQAAAPmQRuRgha2wB2cEg...
Accept: multipart/related; type="application/xml", multipart/related,
multipart/alternative, multipart/batching
Host: ext.vdomain.com:4443
```

4.2.2 HTTP Response

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Via: 1.1 accessproxy.vdomain.com RtcExt
Content-Length: 4145
Content-Type: multipart/related; type="application/xml"; charset=utf-8;
boundary=3c589cb8-79df-4104-a39c-f4e03d2f2993
X-MS-Server-Fqdn: SERVER.vdomain.com
X-MS-Correlation-Id: 2147483653
client-request-id: 270cdbcf-da3a-4230-ad44-e880cfe2f58a
X-MS-Correlation-Id: 2147483680
client-request-id: 4a98d3e3-a561-4a07-acf9-89289788f33d
Strict-Transport-Security: max-age=31536000; includeSubDomains
Date: Wed, 21 Jan 2015 22:43:25 GMT
--3c589cb8-79df-4104-a39c-f4e03d2f2993
Content-Type: application/xml; charset=utf-8
<?xml version="1.0" encoding="utf-8"?>
<events href="/ucwa/oauth/v1/applications/211177894191/events?ack=1"
xmlns="http://schemas.microsoft.com/rtc/2012/03/ucwa">
    <link rel="next" href="/ucwa/oauth/v1/applications/211177894191/events?ack=2" />
    <sender rel="communication"
href="/ucwa/oauth/v1/applications/211177894191/communication">
        <started rel="phoneAudioInvitation"
href="/ucwa/oauth/v1/applications/211177894191/communication/phoneAudioInvitations/aa9
1df7425864b94b25aaf1206f1e795">
            <resource rel="phoneAudioInvitation"
href="/ucwa/oauth/v1/applications/211177894191/communication/phoneAudioInvitations/aa9
1df7425864b94b25aaf1206f1e795">
                <link rel="from"
href="/ucwa/oauth/v1/applications/211177894191/communication/conversations/89938156-
c927-4f1c-a1a2-e99178f0056f/participants/ucwaovbvtuser5@ucwatenant.com"
title="UcwaOVBvtUser5" />
                <link rel="to"
href="/ucwa/oauth/v1/applications/211177894191/people/ucwaovbvtuser2@ucwatenant.com"
/>
                <link rel="cancel"
href="/ucwa/oauth/v1/applications/211177894191/communication/conversations/89938156-
c927-4f1c-a1a2-e99178f0056f/phoneAudio/terminate" />
                <link rel="conversation"
href="/ucwa/oauth/v1/applications/211177894191/communication/conversations/89938156-
c927-4f1c-a1a2-e99178f0056f" />
                <link rel="phoneAudio"
href="/ucwa/oauth/v1/applications/211177894191/communication/conversations/89938156-
c927-4f1c-a1a2-e99178f0056f/phoneAudio" />
                <property name="direction">Outgoing</property><property
name="importance">Normal</property>
                <property name="threadId">AdA1y6o9qIkpVdAlDE6j6U1YN46JpA==</property>
                <property name="state">Connecting</property>
                <property
name="operationId">8eb90e4aa1874134b89dac298d458d20</property>
```

```
                <property name="subject">OV Call</property>
            </resource>
        </started>

    </sender>

</events>
--3c589cb8-79df-4104-a39c-f4e03d2f2993--
```

4.3 Using resync URI

If the server detects that an application is not using the correct request URI to retrieve the event information, it will return the URI that the application can use to get the most recent event information.

In normal operation, applications do not receive a response as described in this example. It usually indicates a design issue in the protocol client. If the protocol client cannot retrieve event information using the URI in the response of this type, it is advisable to discontinue the use of this protocol and report an error to the user so the protocol client can be updated.

The protocol client needs to prevent infinite loops when attempting to retry event information retrievals based on this response.

4.3.1 HTTP Response

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Via: 1.1 accessproxy.vdomain.com RtcExt
Content-Length: 397
Content-Type: multipart/related; type="application/xml"; charset=utf-8;
boundary=819a8d69-873f-4983-a053-761092a07fe5
X-MS-Server-Fqdn: SERVER.vdomain.com
X-MS-Correlation-Id: 2147494723
client-request-id: 095486ef-0b34-47bc-8508-dd1067c13182
X-MS-Correlation-Id: 2147488427
client-request-id: 71adb7b7-0939-4a7e-a4a7-a1b9174ca373
Strict-Transport-Security: max-age=31536000; includeSubDomains
Date: Tue, 27 Jan 2015 21:26:41 GMT
--819a8d69-873f-4983-a053-761092a07fe5
Content-Type: application/xml; charset=utf-8
<?xml version="1.0" encoding="utf-8"?>
<events href="/ucwa/oauth/v1/applications/211618385292/events?ack=999"
    xmlns="http://schemas.microsoft.com/rtc/2012/03/ucwa">
    <link rel="resync" href="/ucwa/oauth/v1/applications/211618385292/events?ack=1" />
</events>
--819a8d69-873f-4983-a053-761092a07fe5--
```

4.4 Error information in event data

For operations requests by the application, errors can be returned through the event channel as shown in the protocol example in this section.

The **reason** element contains additional information about the failure. The **reason** element is an **ErrorType** complex type as described in [MS-OCSMP] section 2.2.4.4.

4.4.1 HTTP Response

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Via: 1.1 accessproxy.vdomain.com RtcExt
```

```
Content-Length: 2593
Content-Type: multipart/related; type="application/xml"; charset=utf-8;
boundary=c22ef9f5-7d38-4738-b58c-33c0198092b8
X-Ms-Namespace: internal
X-MS-Server-Fqdn: SERVER.vdomain.com
X-MS-Correlation-Id: 2147485978
client-request-id: 2935cf79-4d47-452f-9353-3fa339c4ee23
X-MS-Correlation-Id: 2147484766
client-request-id: d3ae0ef7-bb53-4e02-81ab-d991ff2ae56e
Strict-Transport-Security: max-age=31536000; includeSubDomains
Date: Tue, 24 Mar 2015 15:18:25 GMT
--c22ef9f5-7d38-4738-b58c-33c0198092b8
Content-Type: application/xml; charset=utf-8
<?xml version="1.0" encoding="utf-8"?>
<events href="/ucwa/v1/applications/211997960415/events?ack=2"
xmlns="http://schemas.microsoft.com/rtc/2012/03/ucwa">
  <link rel="next" href="/ucwa/v1/applications/211997960415/events?ack=3" />
  <sender rel="conversation"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f">
    <updated rel="phoneAudio"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f/phoneAudio">
      <resource rel="phoneAudio"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f/phoneAudio">
        <link rel="conversation"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f" />
        <property name="state">Disconnected</property>
      </resource>
    </updated>
  </sender>
  <sender rel="communication" href="/ucwa/v1/applications/211997960415/communication">
    <completed rel="phoneAudioInvitation"
href="/ucwa/v1/applications/211997960415/communication/phoneAudioInvitations/bb5bc96fa
8ac4ae3b107458d43c9bab6">
      <status>Failure</status>
      <resource rel="phoneAudioInvitation"
href="/ucwa/v1/applications/211997960415/communication/phoneAudioInvitations/bb5bc96fa
8ac4ae3b107458d43c9bab6">
        <link rel="from"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f/participants/ucwavoiceuser1@ucwatenant.com"
title="UcwaVoiceUser1" />
        <link rel="to"
href="/ucwa/v1/applications/211997960415/people/ucwavoiceuser2@ucwatenant.com" />
        <link rel="conversation"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f" />
        <link rel="phoneAudio"
href="/ucwa/v1/applications/211997960415/communication/conversations/b91f0532-8753-
4ac3-903b-bf348152941f/phoneAudio" />
        <property name="direction">Outgoing</property>
        <property name="importance">Normal</property>
        <property name="threadId">AdBmRcVtEWu8Cs/j9U2epOB9qD8W0w==</property>
        <property name="state">Failed</property>
        <property name="operationId">d7e16f11ea284661aeae43202d3f6e0a</property>
      </resource>
      <reason xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <code>LocalFailure</code>
```

```
        <subcode>PstnCallFailed</subcode>
        <message>The call could not be completed. Please check your number and try
again.</message>
        <parameters />
      </reason>
    </completed>
  </sender>
</events>
--c22ef9f5-7d38-4738-b58c-33c0198092b8--
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full XML Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/rtc/2012/03/ucwa"
    xmlns:tns="http://schemas.microsoft.com/rtc/2012/03/ucwa"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

  <xs:element name="input" type="tns:InputType" />
  <xs:element name="resource" type="tns:ResourceType" />


<xs:element name="events" type="tns:EventsType" />



  <!-- REQUEST-type ELEMENT -->
  <xs:complexType name="InputType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="property" type="tns:PropertyType"/>
      <xs:element name="propertyList" type="tns:CollectionType"/>
    </xs:choice>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- RESPONSE-type ELEMENT -->
  <xs:complexType name="ResourceType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="link" type="tns:LinkType" />
      <xs:element name="property" type="tns:PropertyType" />
      <xs:element name="propertyList" type="tns:CollectionType" />
      <xs:element name="resource" type="tns:EmbeddedResourceType" />
    </xs:choice>

    <!-- The URI of the resource itself -->
    <xs:attribute name="href" type="xs:anyURI" use="required"/>
    <xs:attribute name="rel" type="xs:string" use="optional" />
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- EMBEDDED-RESOURCE ELEMENT -->
  <xs:complexType name="EmbeddedResourceType">
    <xs:complexContent>
      <xs:extension base="tns:ResourceType">
        <!-- The rel of the embedded resource itself -->
        <xs:attribute name="rel" type="xs:string" use="required"/>
        <xs:attribute name="etag" type="xs:string" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- LINK ELEMENT -->
  <xs:complexType name="LinkType">
    <!-- The relationship type of the related resource -->
    <xs:attribute name="rel" type="xs:string" use="required"/>
    <xs:attribute name="href" type="xs:anyURI" use="required"/>
    <xs:attribute name="etag" type="xs:anyURI" use="optional"/>
    <xs:attribute name="title" type="xs:anyURI" use="optional"/>
    <xs:attribute name="revision" type="xs:unsignedByte" use="optional" default="1"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- PROPERTY ELEMENT -->
  <xs:complexType name="PropertyType">
    <xs:simpleContent>
```

```
      <xs:extension base="xs:string">

        <!-- The name of the property -->
        <xs:attribute name="name" type="xs:string" use="required"/>

        <xs:anyAttribute namespace="##other" processContents="lax"/>

      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <!-- COLLECTION (ARRAY, VECTOR) ELEMENT -->
  <xs:complexType name="CollectionType">
    <xs:sequence>
      <xs:element name="item" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>

    <!-- The name of the property -->
    <xs:attribute name="name" type="xs:string" use="required"/>

    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- ERROR ELEMENT -->
  <xs:complexType name="ErrorType">
    <xs:sequence>
      <xs:element name="link" type="tns:LinkType" minOccurs="0" maxOccurs="unbounded" />
      <xs:element name="code" type="xs:string" minOccurs="1" maxOccurs="1" />
      <xs:element name="subcode" type="xs:string" minOccurs="1" maxOccurs="1" />
      <xs:element name="message" type="xs:string" minOccurs="0" maxOccurs="1" />
      <xs:element name="debugInfo" type="tns:ErrorDebugInfoType" minOccurs="0" maxOccurs="1"
/>
      <xs:element name="parameters" type="tns:ErrorParametersType" minOccurs="0"
maxOccurs="1" />
    </xs:sequence>

    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- ERROR PARAMETERS ELEMENT -->
  <xs:complexType name="ErrorParametersType">
    <xs:sequence>
      <xs:element name="property" type="tns:PropertyType" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
  </xs:complexType>

  <!-- ERROR DEBUG INFO ELEMENT -->
  <xs:complexType name="ErrorDebugInfoType">
    <xs:sequence>
      <xs:element name="property" type="tns:PropertyType" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
  </xs:complexType>

 <xs:complexType name="EventType">
    <xs:sequence minOccurs="0">
      <xs:element minOccurs="0" name="in" type="tns:InType" />
      <xs:element minOccurs="0" name="resource" type="tns:ResourceType" />
      <xs:element minOccurs="0" name="reason" type="tns:ErrorType" />
    </xs:sequence>
    <xs:attribute name="rel" type="xs:string" use="required" />
    <xs:attribute name="href" type="xs:string" use="required" />
    <xs:attribute name="title" type="xs:string" use="optional" />
    <xs:attribute name="context" type="xs:string" use="optional" />
  </xs:complexType>

  <xs:complexType name="InType" >
    <xs:attribute name="rel" type="xs:string" use="required" />
```

```
      <xs:attribute name="href" type="xs:string" use="required" />
      <xs:attribute name="title" type="xs:string" use="required" />
    </xs:complexType>

    <xs:complexType name="SenderType">
      <xs:sequence>
        <xs:choice maxOccurs="unbounded">
          <xs:element name="added" type="tns:EventType" />
          <xs:element name="updated" type="tns:EventType" />
          <xs:element name="deleted" type="tns:EventType" />
          <xs:element name="started" type="tns:EventType" />
          <xs:element name="completed" type="tns:EventType" />
        </xs:choice>
      </xs:sequence>
      <xs:attribute name="rel" type="xs:string" use="required" />
      <xs:attribute name="href" type="xs:string" use="required" />
    </xs:complexType>

    <xs:complexType name="EventsType" >
      <xs:sequence>
        <xs:element name="link" type="tns:LinkType" minOccurs="1" maxOccurs="1" />
        <xs:element maxOccurs="unbounded" name="sender" type="SenderType" />
      </xs:sequence>
      <xs:attribute name="href" type="xs:string" use="required" />
    </xs:complexType>

  </xs:schema>
```

7 Appendix B: Product Behavior

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

- Microsoft Lync Client 2013/Skype for Business
- Microsoft Skype for Business 2016
- Microsoft Skype for Business Server 2015

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

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

8 Change Tracking

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
2.2.1 Namespaces	Added tns prefix.	Y	Content update.
2.2.4.1 EventsType	Added descriptions for the elements and attributes.	N	Content update.
2.2.4.2 EventType	Added descriptions for the elements and attributes.	N	Content update.

9 Index

T

U

V

X