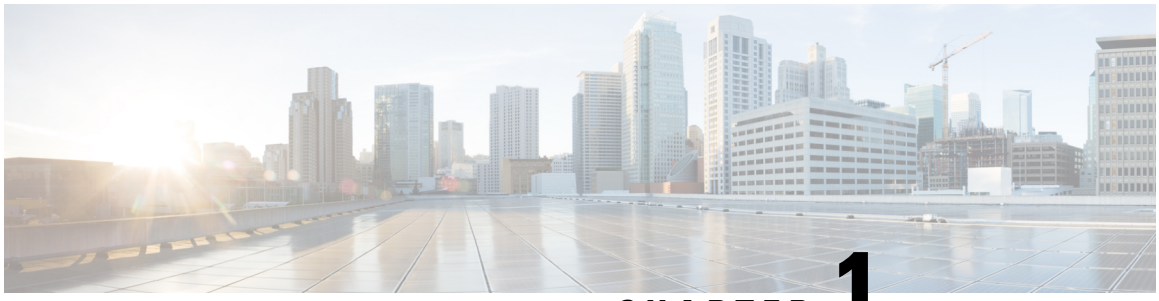
Cisco Unity Connection Notification Interface (CUNI) API

First Published: 2017-02-02

Last Modified: 2022-05-12

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS



CHAPTER 1

API Overview

Introduction

The Cisco Unity Connection Notification Interface (CUNI) API is a web service interface for managing subscriptions to events from Cisco Unity Connection systems. It provides a way to get asynchronous notifications when voice mail messages are received, deleted, or changed. CUNI can be used to integrate Connection into an existing enterprise-wide portal.

CUNI is designed to provide a simple, stable method of accessing the notification functionality on Connection systems through a standards based interface using XML and HTTPS.

Through CUNI, you can do the following:

- Subscribe for message events
- Subscribe for one or many users on a single channel

Note: All the above functions associated with CUNI API support both the IPv4 and IPv6 addresses. However, the IPv6 address works only when Connection platform is configured in Dual (IPv4/IPv6) mode.

Note that CUNI is intended for use in server-to-server applications where receiving notifications for many users over a single connection is required. As such, it is designed to handle a small number of clients that are each subscribing for notifications on a large set of subscribers. It also requires Administrative credentials, making it inappropriate for browser applications to use directly.

CUNI is composed of two parts: a SOAP interface for subscribing, and an asynchronous piece (the Notifier) that delivers events as XML over HTTP.

Technical Details

- CUNI is standards-based: it implements a standard SOAP-based interface for managing subscriptions. Message events are sent as standard XML over HTTP.

- CUNI is easy to use: as a web-based interface, CUNI is independent of operating systems and programming languages, and does not require any client libraries to use.

Getting Started with CUNI

In order to begin developing with CUNI, you need to obtain the following:

Hardware

- Cisco Media Convergence Server (MCS) for Cisco Unity Connection.
- For detailed hardware-specification information, see the Supported Platforms List available at https://www.cisco.com/c/en/us/support/unified-communications/unity-connection/products-installation-guides-list.html .

Software

- Cisco Unity Connection Software Ordering
- Not for Resale Kits (Must for eligible to purchase)
- Select Unified Communications System Release Kit

Discounts for some of the above hardware and software may be available for participants in the Cisco Technology Developer Program.

We recommend that all developers have an up-to-date Cisco Developer Services support agreement. This provides the developer with access to professional support and assistance for application development.

Other CUNI Resources

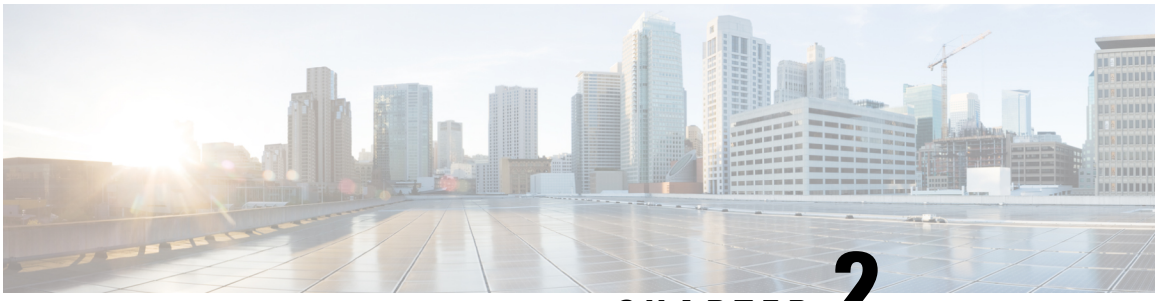
Additional information about CUNI is also available on the Cisco Developer Network, see the https://developer.cisco.com/site/unity-connection/overview/.

To participate in the Unity Connection forum, see the https://communities.cisco.com/community/developer/collaboration/voicemail.

Troubleshooting

See the following for information on troubleshooting all Connection APIs:

https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/connection/REST-API/APIs_Pages/b_CUC_API_troubleshooting.html



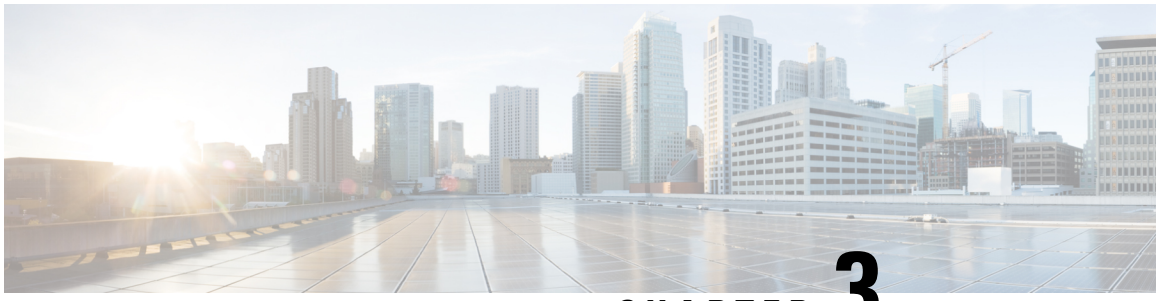
CHAPTER 2

Cisco Unity Connection Notification Interface (CUNI) API -- CUNI Event Schema

Schema Example

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns="\[http://www.cisco.com"\] elementFormDefault="qualified"
targetNamespace="\[http://www.cisco.com"\] xmlns:xs="\[http://www.w3.org/2001/XMLSchema">\]
<xs:simpleType name="priorityType">
<xs:restriction base="xs:string">
<xs:enumeration value="Low-Priority" />
<xs:enumeration value="Normal-Priority" />
<xs:enumeration value="Urgent" />
<xs:enumeration value="Unknown-Priority" />
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="eventType">
<xs:restriction base="xs:string">
<xs:enumeration value="MESSAGE_INFO" />
<xs:enumeration value="NEW_MESSAGE" />
<xs:enumeration value="SAVED_MESSAGE" />
<xs:enumeration value="UNREAD_MESSAGE" />
<xs:enumeration value="DELETED_MESSAGE" />
<xs:enumeration value="FAILOVER" />
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="messageType">
<xs:restriction base="xs:string">
<xs:enumeration value="Voice" />
<xs:enumeration value="NDR" />
<xs:enumeration value="DR" />
<xs:enumeration value="RR" />
<xs:enumeration value="Fax" />
<xs:enumeration value="Text" />
<xs:enumeration value="UnknownType" />
</xs:restriction>
</xs:simpleType>
<xs:complexType name="messageInfoType">
<xs:attribute name="messageId" type="xs:string" />
<xs:attribute name="receiveTime" type="xs:string" />
<xs:attribute name="msgType" type="messageType" />
<xs:attribute name="uid" type="xs:integer" />
<xs:attribute name="priority" type="priorityType" />
```

```
<xs:attribute name="sender" type="xs:string" />
<xs:attribute name="callerAni" type="xs:string" />
</xs:complexType>
<xs:element name="messageEvent">
<xs:complexType>
<xs:sequence minOccurs="1" maxOccurs="unbounded">
<xs:element name="messageInfo" type="messageInfoType" />
</xs:sequence>
<xs:attribute name="subscriptionId" type="xs:string" />
<xs:attribute name="eventType" type="eventType" />
<xs:attribute name="eventTime" type="xs:string" />
<xs:attribute name="mailboxId" type="xs:string" />
<xs:attribute name="displayName" type="xs:string" />
<xs:attribute name="USN" type="xs:integer" />
</xs:complexType>
</xs:element>
</xs:schema>
```



CHAPTER 3

Cisco Unity Connection Notification Interface (CUNI) API -- Subscribing to and Processing Notification Events

Subscribing to Notification Events

Subscribing to notification events is done through the web service interface, and can be as simple as a single web service call to "subscribe".

At a minimum you should pass in:

- The callback URL where XML notifications will be posted by the Notifier.
- The date/time when the subscription will expire

It is also possible to pass in a list of the resources (userid) that you are interested in receiving events for, although that can also be done via a subsequent call to "addResourceIdToSubscription". We recommend that you leave the eventTypeList empty, indicating that you would like to receive all types of message events.

This example shows making a call to subscribe in Java, using Axis2 stubs generated from the WSDL:

```
Calendar expiration = java.util.Calendar.getInstance();
com.cisco.unity.messageeventservice.MessageEventServiceStub.Subscribe s =
    new com.cisco.unity.messageeventservice.MessageEventServiceStub.Subscribe();

// Set subscription expiry to two months from now.
expiration.add(Calendar.MONTH, 2);
s.setExpiration(expiration);

// Specify the users we're interested in, which is just the operator for this example.
ArrayOfString resourceList = new ArrayOfString();
resourceList.addString("operator");

s.setResourceIdList(resourceList);
s.setExpiration(expiration);

// Set the callback information - a username and password can be specified, but they are
optional.
MessageEventServiceStub.CallbackServiceInfo c = new
MessageEventServiceStub.CallbackServiceInfo();
c.setCallbackServiceUrl(callbackUrl);
c.setPassword(callbackPassword);
c.setUsername(callbackUserId);
s.setCallbackServiceInfo(c);

// Subscribe!
SubscribeResponse r = stub.subscribe(s);
```

Note CUNI Subscriptions will be removed from Cisco Unity Connection server database, if you perform a refresh upgrade. Make sure to perform re-subscription after successful upgrade of the cluster.

Processing Notification Events

Events will be delivered via HTTP POST to the callback URL that was provided in the call to subscribe.

Note Cisco Unity Connection Release 12.5(1) SU3 and later, CUNI supports HTTPS for callback URL.

For SSL communication user has the option to choose between Self-signed Certificates and Third Party CA signed Certificates from Cisco Unified Mobility Advantage server. Make sure to follow below mentioned steps for Third Party CA signed certificates:

1. Upload Third Party CA signed certificates in tomcat trust of Unity Connection server.
2. Restart Tomcat and Notifier services after adding certificates.

Event Type	Description
NEW_MESSAGE	This event is sent when a new message arrives in the Inbox folder.
SAVED_MESSAGE	This event is sent when a message is marked as read in the Inbox folder.

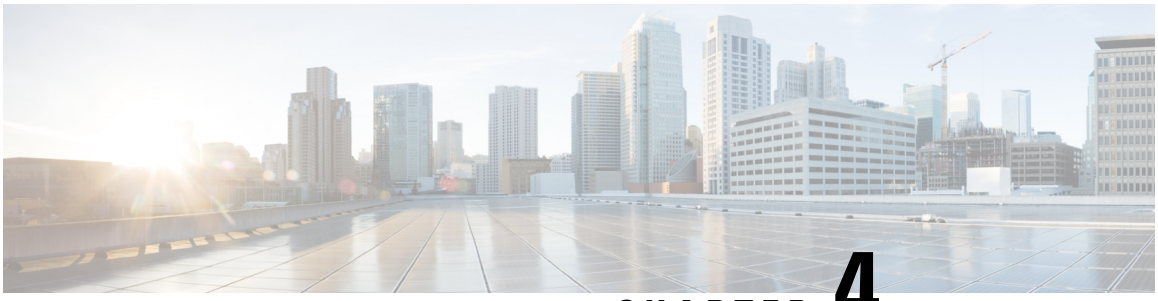
Event Type	Description
UNREAD_MESSAGE	This event is sent when a message is marked as unread in the Inbox folder.
DELETED_MESSAGE	This event is sent when a message is deleted from the Inbox folder.
DELETED_MESSAGE_DELETE	This event is sent when a message is hard deleted from the Deleted folder or from the Inbox folder.
DELETED_MESSAGE_CREATE	This event is sent when a new message arrives in the Deleted folder.
DELETED_MESSAGE_READ	This event is sent when a message is marked as read in the Deleted folder.
DELETED_MESSAGE_UNREAD	This event is sent when a message is marked as unread in the Deleted folder.
DELETED_MESSAGE_UNDELETE	This event is sent when a message is marked as undeleted from the Deleted folder
TRANSCRIPTION_NEW_MESSAGE	This event is sent when transcribed message arrives on web clients.

Example

This is an example of an event xml. Please note that it is valid to have more than one messageInfo per messageEvent:

```
<?xml version="1.0" ?>
<messageEvent subscriptionId="00bdcfd3-159a-48d3-ac7b-2f3b4f83db6c"
              eventType="SAVED_MESSAGE"
              eventTime="11:15:40 PM 10/30/2008"
              mailboxId="abell"
              displayName="Alexander Bell"
              USN="2265">
    <messageInfo messageId="72204bd7-e5c3-446e-adb6-ae5f80db26fb"
                 receiveTime="04:15:40 PM 10/30/2008"
                 uid="543"
                 msgType="Voice"
                 priority="Normal-Priority"
                 sender="null"
                 callerAni="null" />
</messageEvent>
```

After receiving notifications from CUNI API, if user want to perform messaging operations, then refer "Inbox Folder Operations" section of "Cisco Unity Connection Messaging Interface (CUMI) API -- Using the CUMI API" chapter of *Cisco Unity Connection Messaging Interface (CUMI) API Guide* available at https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/connection/REST-API/CUMI_API/b_CUMI-API.html



CHAPTER 4

Cisco Unity Connection Notification Interface (CUNI) API -- CUNI FAQs

CUNI FAQs

- **Where Can I Get the WSDL?**

  The WSDL can be found at:
  http://<connection-server>/messageeventservice/services/MessageEventService?wsdl

  Note The WSDL is the same across versions except that for 8.0 there was a keepAlive parameter added to the subscribe method.

- **What Happens to Subscriptions if the Cisco Unity Connection Server Is Restarted?**

  Subscriptions are stored in the database. When Cisco Unity Connection is restarted, the Notifier reads the existing subscriptions and begins sending out notifications.

- **What is authentication mechanism to register for CUNI events?**

  Basic AUTH/HTTPS – requires administrative credentials.

- **Can several listeners subscribe for the events for same mailbox specifying different callback urls?**

  Yes they can

- **Does CUNI support HTTPS for subscribing for events as well can a callback url be https?**

  Yes, Cisco Unity Connection Release 14, 12.5(1) SU3 and later CUNI supports HTTPS for both subscribing and callback url.

- **Does CUNI in HTTPS mode support Multi SAN CA signed certificates configured on callback?**

  No, CUNI in HTTPS mode only supports self signed and CA signed Certificates.

- **What kind of event or alerting mechanism is used in case it cannot reach the callback url? Any SNMP alert?**

No there are no SNMP alerts generated by the CUNI. There are traces that log information on the Unity Connection server.

- **What happens to events in case of server reboot? Are they queued up and sent to the listeners when it boots up?**

  No they are not queued up. They will be lost of a reboot.

- **Are subscriptions synchronized between sub and pub? What happens in case one goes down?**

  Yes they are synchronized so they pub can pick them and take them over.

- **Are There Any Unsupported Methods?**

  Yes. subscribeForAllResources is currently not supported, so users must be listed explicitly. Also there is no current support for SSL connections for callbacks.

- **Can Multiple Subscriptions Use the Same Callback URL?**

  Not exactly. If Cisco Unity Connection sees a repeat subscription for a callback URL, it will delete the previous subscription as a safeguard against ending up with unintentional subscriptions.

- **Is CUNI a replacement for COMET Notifications?**

  No. Comet notifications are primarily for sending notifications to browser clients, each representing a single user. CUNI is not a replacement for comet: it is for server-to-server notifications, whereas Comet is for server-to-client notifications.

- **What is the main use case that CUNI is intended for? What are the advantages over comet?**

  CUNI is meant for sending message events for multiple users over a single channel, such as in a server-to-server applications.

- **How does CUNI scale for about 20,000 users? Does it have any scale limitations in terms of number of subscriptions?**

  CUNI has been tested with several thousand users. Suggestion is for using one limiting per subscription to 500 user at maximum. Each subscription can register to receive events for multiple users.

- **How is the event notification made to the listener, are several events batched and sent at a set interval or are they sent as they occur?**

  Notifications are mainly sent as they occur. If a single user has a bunch in rapid succession, they will be batched.