

Enhanced RTP Streaming XML Service Interface Application Note



Target Audience: Unified Communication XML Application developers

Concept: This document was created by the Cisco Voice Technology Group as part of a series of documents to provide information and implementation guidance on new XML application interface objects as they become available.

The XML Service Interface documentation and SDK are updated as new versions of Cisco Unified Communication Manager are released. In between releases of Cisco Unified Communication Manager, new releases of IP phone firmware are also released. Some of these IP phone firmware releases may contain enhancements to the XML Service Interface. This series of XML service interface applications notes has been produced as a way of making these enhancements available to the developer community as early as possible.

The information contained in this application note will be incorporated into the next release of the XML application interface documentation and the SDK. At that time, the information in this document will be deprecated, but until that time this document acts as the reference documentation for the XML Service Interface API being described.



Enhanced RTP Streaming

Table of Contents

Introduction	2
Syntax	3
RTP Streaming Schema	3
Error Schema.....	6
Sample Usage	6
Pre-requisites	8
Supported Platforms	8
Error and Response.....	8
Debug / Logging.....	9
Caveats	9
Additional References	9

Introduction

This XML-based RTP streaming API allows applications to initiate and observe RTP audio streams. It provides extended capabilities beyond what can be done with the legacy RTP streaming URIs with support for stream start/stop event listeners and the ability to specify other extended stream attributes, such as codec type.

The event handlers will most commonly use the standard Notification framework (Notify URI), but they can also invoke most other URIs, with the exception of HTTP URLs.

The “port” number parameter of the “startMedia” request is optional and if not specified, the phone will pick an available port and return it in the “startMediaResponse” object. The “port” parameter, if specified, must be an even number in the range 20480-32768.

This new XML-based RTP API allows a full-duplex stream (mode=sendReceive) to be setup as a single stream request which simplifies the usage of the API, but creates some interoperability issues with the legacy RTP URIs because they treat send and receive streams separately. The interaction rules between legacy RTP URI streams and the new RTP XML API are as follows:

- If an RTP Stop URI is invoked, and an RTP XML API stream is currently streaming in that same direction, then the **entire** RTP XML API stream is stopped.
For example, if a full-duplex stream is setup thru the RTP XML API (mode=sendReceive) and then an RTPTx:Stop URI is invoked, the stream will be stopped in both the send and receive directions (and the onStopped event handler will be



called, if present).

- If the RTP XML “stopMedia” request does **not** specify a stream “id”, then the request will stop **all** services RTP streams, in any direction (send or receive) and of any type (multicast and unicast). This allows applications using the new RTP XML API to stop media streams which may have been started by the legacy RTP URIs or by other applications for which a stream ID is not known.

Syntax

RTP Streaming Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Cisco
Systems, Inc. (Cisco Systems, Inc.) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="startMedia">
    <xs:complexType>
      <xs:all>
        <xs:element name="mediaStream" type="mediaStream"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="stopMedia">
    <xs:complexType>
      <xs:all>
        <xs:element name="mediaStream">
          <xs:complexType>
            <xs:attribute name="id" type="xs:string"
use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:all>
    </xs:complexType>
  </xs:element>
  <xs:element name="startMediaResponse">
    <xs:complexType>
      <xs:all>
        <xs:element name="mediaStream" type="mediaStream"/>
      </xs:all>
    </xs:complexType>
  </xs:element>
</xs:schema>
```



```
</xs:complexType>
</xs:element>
<xs:element name="notifyMediaEvent">
  <xs:complexType>
    <xs:all>
      <xs:element name="mediaStream">
        <xs:complexType>
          <xs:attribute name="id" type="xs:string"
use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:all>
    <xs:attribute name="origin" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="user"/>
          <xs:enumeration value="application"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="stopped"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:complexType name="mediaStream">
  <xs:all>
    <xs:element name="type">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="audio"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="codec">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="G.711"/>
          <xs:enumeration value="G.722"/>
          <xs:enumeration value="G.723"/>
          <xs:enumeration value="G.728"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:all>
</xs:complexType>
</xs:element>
```



```
<xs:enumeration value="G.729"/>
<xs:enumeration value="GSM"/>
<xs:enumeration value="Wideband"/>
<xs:enumeration value="iLBC"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="mode">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="send"/>
      <xs:enumeration value="receive"/>
      <xs:enumeration value="sendReceive"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="address">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="7"/>
      <xs:maxLength value="15"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name="port" minOccurs="0">
  <xs:simpleType>
    <xs:restriction base="xs:unsignedShort">
      <xs:minInclusive value="20480"/>
      <xs:maxInclusive value="32768"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
</xs:all>
<xs:attribute name="onStopped" use="optional">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="1"/>
      <xs:maxLength value="256"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
<xs:attribute name="receiveVolume" use="optional">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
```



```
        <xs:maxInclusive value="100"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
</xs:schema>
```

Error Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="errorResponse">
    <xs:complexType>
      <xs:all>
        <xs:element name="type">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="InvalidURL"/>
              <xs:enumeration value="InvalidResource"/>
              <xs:enumeration value="InvalidResourceID"/>
              <xs:enumeration value="UnavailableResource"/>
              <xs:enumeration value="InvalidXML"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="data" nillable="true">
          <xs:simpleType>
            <xs:restriction base="xs:string"/>
          </xs:simpleType>
        </xs:element>
      </xs:all>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Sample Usage

Start Media

Request

HTTP POST /CGI/Execute



```
<startMedia>
  <mediaStream
    onStoped="Notify:http:server:80:path/page"
    receiveVolume="50">
    <type>audio</type>
    <codec>G.729</codec>
    <mode>sendReceive</mode>
    <address>239.1.2.3</address>
    <port>20480</port>
  </mediaStream>
</startMedia>
```

Response

```
HTTP200 OK
<mediaStream id="abc123"/>
```

Stop Media

Request

```
HTTP POST CGI/Execute
<stopMedia>
  <mediaStream id="abc123"/>
</stopMedia>
```

Response

```
HTTP 200 OK
{no body}
```

Media Stopped Notification

For the <mediaStream> above, if the user terminated the media stream by placing the active audio path on-hook, the following notification would be sent

```
HTTP POST /server/path/page
DATA=<notifyMediaEvent type="stopped" origin="user">
  <mediaStream id="abc123"/>
</notifyMediaEvent>
```



Pre-requisites

This XML Service Interface API, requires IP phone firmware 8.3(2) or later. For the availability of the 8.3(2) phone firmware, please consult www.cisco.com or contact the Developer Support team in order to obtain an early release of this phone firmware.

http://www.cisco.com/en/US/products/svcs/ps3034/ps5408/ps5418/serv_home.html

NOTE: Phone Firmware 8.3(2) contains an updated XML parser, with more rigid enforcement of the XML schema. XML applications that have been written to earlier versions of the XML parser, may encounter exceptions if they do not conform to the XML schema. It is recommended that any existing XML applications be validated with the new XML parser before trying to enhance the XML application to use this new XML Service Interface API.

For details on the new XML parser, please consult the XML Service Interface Application note entitled **IP Phone Services XML Schema Enforcement**

Supported Platforms

The following models of IP phones will support this XML Service Interface API, 7906, 7911, 7931, 7941, 7961, 7970, and 7971. Although several codecs are listed within the schema, only the codecs G711, G729 and G722 are currently supported.

The 7905, 7912, 7920, 7940, 7960 and 7985 will not support this XML Service Interface API

Error and Response

If any of these API request fails, then an appropriate HTTP 4xx error code is returned, and the HTTP body is a text/xml payload which contains an <errorResponse> object. See the XML schema for the details, but here is a sample error response:

HTTP 400 Bad Request

```
<errorResponse>
  <type>InvalidResourceID</type>
  <data>Unknown Media Stream ID: abc123</data>
</errorResponse>
```

The following table lists the possible error conditions are their respective error codes and descriptions

Condition	Applicable Methods	HTTP Result Code	Type	Data
Authorization failed	<all>	401 (Auth Failed)	N/A	N/A
Request object	<all>	400 (Bad Request)	InvalidXML	{parser error



does not comply with the API's XML schema				description}
Media cannot be started because there's no DSP resource available to handle the media	startMedia	400 (Bad Request)	UnavailableResource	No Media Resource Available
Media cannot be stopped because the specified stream ID doesn't exist	stopMedia	400 (Bad Request)	InvalidResourceID	Unknown Media Stream ID: {stream ID}

Debug / Logging

Any failures to deliver a notification will be logged to the phone console (and web console logs) to allow debugging of undelivered events.

Caveats

Although several codecs are listed within the schema, only the codecs G711, G729 and G722 are currently supported. The RTP stream API is intended for use with an idle phone, and is not intended for use during an active call. If an attempt is made to instruct the IP phone to stream audio during an active call, the instruction will be denied by the phone.

Additional References

Cisco IP phone Services along with developer resources such as the SDK to assist developers in getting started with applications and documentation on the various URIs and XML Objects that are supported in the IP phones are available at the link posted below.

http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_usage_guide_book09186a00807a34b9.html

In recent years, Cisco has introduced a new generation of desktop IP phone models, the 7906, 7911, 7931, 7941, 7961, 7970 and 7971. These new models of phones have been introduced as a platform in which new features can be introduced, and on which new applications can be built. The introduction of these new generation IP phones has also provided an opportunity to increase the range of XML objects and URIs that are supported. The introduction of enhancements to IP phone services is also the time to remind application developers, of the XML schema that is defined for the IP phone services. This schema is published at the following location, and has been since initial release



http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_usage_guide_chapter09186a00807a352c.html

Please contact the developer support team and gain more information about Developer services and access to early deployment phone loads.

http://www.cisco.com/en/US/products/svcs/ps3034/ps5408/ps5418/serv_home.html