



Cisco Unified Workforce Optimization

Quality Management API Programmers Guide 2.6
September 2008

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CCDE, CCENT, Cisco Eos, Cisco Lumin, Cisco Nexus, Cisco StadiumVision, Cisco TelePresence, the Cisco logo, DCE, and Welcome to the Human Network are trademarks; Changing the Way We Work, Live, Play, and Learn and Cisco Store are service marks; and Access Registrar, Aironet, AsyncOS, Bringing the Meeting To You, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, CCVP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unity, Collaboration Without Limitation, EtherFast, EtherSwitch, Event Center, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, iQuick Study, IronPort, the IronPort logo, LightStream, Linksys, MediaTone, MeetingPlace, MeetingPlace Chime Sound, MGX, Networkers, Networking Academy, Network Registrar, PCNow, PIX, PowerPanels, ProConnect, ScriptShare, SenderBase, SMARTnet, Spectrum Expert, StackWise, The Fastest Way to Increase Your Internet Quotient, TransPath, WebEx, and the WebEx logo are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or Website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0807R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

Quality Management API Programmers Guide

© 2008 Cisco Systems, Inc. All rights reserved.

© 2008 Calabrio, Inc. All rights reserved.

Contents

Quality Management Recording API 5

- Introduction 5
- Overview 6
 - API Commands and Network Recording 6
- API Commands 8
 - Delete Command 8
 - Metadata Command 8
 - Pause Command 10
 - Restart Command 10
 - Resume Command 10
 - Tag Command 11
- Integration with CAD 12
 - Integrating with the QM Recording API Commands 12
 - Integrating with the Metadata Command 13

Quality Management Server API 17

- Introduction 17
- Data Model 18
 - SOAP Binding 18
 - Character Encoding 18
 - Binding Style 18
 - Transport Protocols 18
 - SOAP Action Header 18
 - Port 18
 - SOAP Header 18
 - Request and Response Messages 18
 - Fault Messages 19
 - Namespaces 19
 - Downloading WSDL and XML Schema Files 19
- Authentication 20
 - Including Authentication Information 20

Contents

Example of a SOAP Header	20
User Scope	20
■ API Operations	21
Search Operation	21
Request	21
Response	21
Paging	22
Export Recording Operation	22
Request	22
Response	23
Downloading the Exported Recording	23
Export Progress Operation	23
Request	23
Response	24
Delete Recording Operation	24
Request	24
Response	25
Edit Metadata Operation	25
Request	25
Response	26

Introduction

The Quality Management (QM) Recording API provides a means for users to create an external application that interfaces with the QM system and enables agents to perform the following actions:

- Tag calls for recording and retention
- Pause a recording
- Resume a recording
- Restart a recording
- Delete calls marked for recording
- Attach metadata to calls.

The QM Desktop Recording service and the QM Network Recording service enable the API by accepting formatted requests passed via sockets to an IP address and port.

Cisco Agent Desktop (CAD) can integrate with QM via the QM Recording API. CAD does this via its interprocess communication (IPC) action. IPC actions pass information in the form of user datagram protocol (UDP) messages from the agent desktop to a third-party application (in this case, the QM Recording API) using IPC methods.

Overview

Following are definitions of terms used in describing API commands.

- **Active Call.** The call that is currently connected.
- **Last Call.** The previous active call that is valid only up to the time when the next call begins. The last call before the configured End of Day, user logout, service shutdown, or CTI service disconnect will not be counted as the last call.

API commands are case-insensitive. The following three statements are considered identical.

```
command=tag
command=TAG
COMMAND=Tag
```

Multiple API commands can be made for the same call. For example, a call can have metadata attached to it and be tagged for retention. However, once a call is deleted using the Delete command, the Metadata and Tag commands will have no effect.

Each API command must be sent in its own message. You cannot send multiple API commands in a single message.

No headers are required.

The commands are “one way.” That is, no confirmation of success or failure will be returned when they are invoked.

Connection information is as follows:

```
Protocol:    UDP
IP Address:  127.0.0.1 (the “localhost” IP address)
Port:       65001
```

API Commands and Network Recording

All API commands work with the QM network recording functionality.

Connection information is as follows:

```
Protocol:    UDP
IP Address:  The IP address or hostname of the QM Network Recording
             service host machine
Port:       65002
```

Since there can be multiple QM Network Recording service instances, you must know which QM Network Recording service a specific user is associated with in order to specify the correct IP address/hostname to which to send the API command. Information about this association can be found in QM Administrator (Record Server Configuration > VoIP Devices node).

The keyword **sender_id** must be added to any API command intended for a network recording user. The sender_id is the User ID associated with the user as found in QM Administrator (Personnel > User Administration node).

NOTE: API commands intended for an endpoint recording user (users whose PCs have the QM Desktop Recording service installed) must NOT contain the sender_id keyword.

For example, to use the Tag command to tag a network recording for a particular agent, the command is formatted as follows:

```
command=tag&sender_id=5000.1234
```

Where 5000.1234 is the agent's User ID.

The sender_id keyword must always follow immediately after the API command keyword and before any additional data, such as metadata. For example:

```
command=metadata&sender_id=5000.1234&key1=value1&key2=value2
```

API Commands

This section describes the commands available in the QM Recording API.

Delete Command

`command=delete`

The Delete command is used to mark a recording to be deleted, even if archiving is enabled, the call meets workflow criteria, the extension is in the inclusion list, or it has been tagged by the agent for retention. The recorded files and any metadata will be deleted, but the basic contact data will still be uploaded to the system in order to maintain accurate call counts.

- The Delete command is valid for the active call only.
- The Delete command has precedence over all other commands.
- Calls that are deleted are not available for archive or quality management purposes.
- Calls that are deleted are not viewable in QM Desktop.

Metadata Command

`command=metadata&key1=value1&key2=value2`

The Metadata command enables metadata to be attached to the active call. If the call is uploaded because archiving is enabled or it matches workflow criteria, then the metadata is uploaded as well. If the call is not uploaded, then the metadata is discarded.

- The Metadata command is valid for the active call and the last call.
- A maximum of 10 metadata items can be associated with a call. This can be accomplished with 10 Metadata commands containing one key/value pair each, or one Metadata command containing up to 10 key/value pairs.
- Only metadata that has been defined in QM Administrator (Enterprise > Recordings > Metadata) can be attached to a call. If an unknown key is added to a Metadata command, it is ignored.

The Metadata command interacts with the active call, including the time up until the next call starts. If the command is invoked during the call, the metadata is uploaded to the database at the same time as the rest of the call data. If the command is invoked after the call but before the next call, the metadata is uploaded separately at the time the command is invoked, and will be stored with the last known call.

NOTE: The last known call is reset at logon, so metadata cannot be attached to the last known call before logout or shutdown after the

next logon occurs. Metadata will be attached to calls that span the configured end of day/upload time.

Successive calls to the Metadata command using the same key name will update the existing metadata for that call. For example:

Original metadata	&custID=1000&balance=500
Updated metadata	&balance=200
Final metadata	&custID=1000&balance=200

Specifying an empty value for a key ("key=") zeros out existing metadata for that key. For example:

Original metadata	&custID=1000&balance=500
Updated metadata	&balance=
Final metadata	&custID=1000&balance=

Valid formats for metadata are as follows.

Dates	Dates must be in yyyy-mm-dd format. Valid date: 2008-04-19
Numbers	Numbers can start with and contain a decimal point. They cannot end with a decimal point or contain a comma. Valid numbers: .30 10.7 2500 Invalid numbers: 30. 2,500
Text	Key data and value data cannot contain the following reserved characters. All other alphanumeric characters are valid. Reserved characters: & =

The QM Recording client reloads configuration data (that is, workflow and settings) after the first call that ends after the configured end of day/upload time, and if the agent logs out and logs in again, so if the metadata fields were changed in QM Administrator, any third-party application that is supplying metadata to the QM Recording client might need to be updated to reflect any changes.

Pause Command

command=pause

The Pause command is used to temporarily halt recording the audio portion of a contact.

- The Pause command does not affect the video portion of the contact.
- The Pause command is valid for active calls only.
- If a call is already paused, the Pause command has no affect.
- When a recording that was paused is played back, the video portion of the recording continues to play during the paused audio portion.

Restart Command

command=restart

The Restart command is used to restart or start recording a call.

- If an active call is currently being recorded, the Restart command stops the audio and video recording, deletes that recording, and starts recording the call from the point that the Restart command was invoked.
- If an active call is not currently being recorded, the Restart command starts audio and video recording.
- The Restart command is valid for active calls only.
- Calls that are recorded using the Restart command are given the Agent Tagged reason code. The agent tagged calls are saved even if archiving is not enabled and the call does not meet workflow criteria.

Resume Command

command=resume

The Resume command is used to resume recording the audio portion of a contact after recording has been paused using the Pause command.

- If the call is not currently paused, the Resume command has no affect.
- The Resume command does not affect video recording.
- The Resume command is valid for active calls only.
- If the Resume command is not used, the point at which a recording is paused is the end of the audio recording.

Tag Command

command=tag

The Tag command is used to mark a recording to be saved, even if archiving is not enabled and the call does not meet workflow criteria. Agent-tagged calls are stored with the Agent Tagged reason code, and are saved for the retention time configured in QM Administrator.

- The Tag command is valid for the active call and the last call.
- If the active call is already being recorded, the Agent Tagged reason code is added to the data associated with the call.
- If the active call is not already being recorded, recording is started when the command is invoked and the Agent Tagged reason code is added.
- If the last call was recorded, the reason code is updated to the Agent Tagged reason.
- If the last call was not recorded, nothing happens. There is no recording available whose reason code can be updated.
- If there are two active calls (for example, an inbound ACD call and an outbound consultation call) being recorded, whichever call triggered recording to begin will be tagged.
- If there are two active calls (for example, an inbound ACD call and an outbound consultation call) and neither is currently being recorded, the call that was delivered to the agent first, based on the call start times, will begin recording and be tagged when the Tag command is invoked.

Integration with CAD

Cisco Agent Desktop (CAD) can use its IPC Action feature to integrate with Quality Management Desktop via the QM Recording API.

Cisco Agent Desktop (CAD) can integrate with QM via the QM Recording API. CAD does this via its interprocess communication (IPC) action. IPC (interprocess communication) actions pass information in the form of UDP messages from the agent desktop to the QM Recording API using IPC methods.

Integrating with the QM Recording API Commands

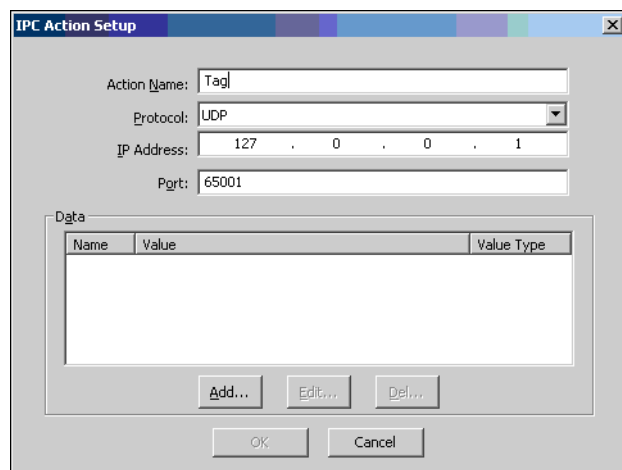
The following example shows how to configure an IPC action to invoke the Tag command. The other QM Recording API commands can be invoked in this way. An exception is the Metadata command. See ["Integrating with the Metadata Command" on page 13](#) for more information.

For more explicit information on creating and configuring actions, see the *Cisco Desktop Administrator User Guide*.

To configure an IPC Action to invoke the Tag command:

1. In Cisco Desktop Administrator, create a new IPC action. In the IPC Action Setup dialog box, complete the fields as follows:
 - Action Name: Tag
 - Protocol: UDP
 - IP Address: 127.0.0.1 (the "localhost" IP address)
 - Port: 65001

Figure 1. IPC Action Setup dialog box.



- In the Data section, click Add to display the IPC Request Data Dialog box.

Figure 2. IPC Request Data Dialog box.

- Complete the fields as follows and then click OK to close the dialog box.
 - Value Name: command
 - Value Type: UserDefined
 - Value: tag
- Click OK to save the IPC Action.

Integrating with the Metadata Command

The Metadata command can be integrated with CAD, but it requires more setup than do the other API commands.

Before configuring an IPC action in CAD, configure the user-defined metadata in Quality Management Administrator. See the *Quality Management Administrator User Guide* for more information. For the purposes of this example, the following metadata is configured as shown in [Figure 3](#):

- Balance
- CardType
- CustomerID

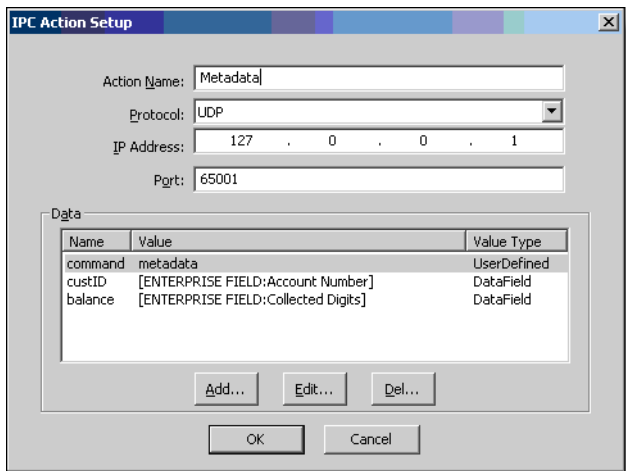
Figure 3. Metadata configured in QM Administrator

Display Name	Key Name	Type	Encrypted
Balance	balance	Number	<input checked="" type="checkbox"/>
CardType	cardType	String	<input checked="" type="checkbox"/>
CustomerID	custID	Number	<input checked="" type="checkbox"/>

To configure an IPC Action to invoke the Metadata command:

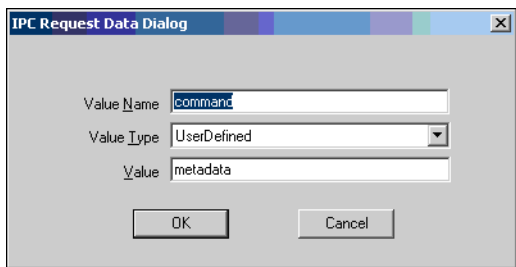
1. In Cisco Desktop Administrator, create a new IPC action. In the IPC Action Setup dialog box, complete the fields as follows:
 - Action Name: metadata
 - Protocol: UDP
 - IP Address: 127.0.0.1 (the “localhost” IP address)
 - Port: 65001

Figure 4. IPC Action Setup dialog box.



2. In the Data section, click Add to display the IPC Request Data Dialog box.

Figure 5. IPC Request Data Dialog box.



3. Complete the fields as follows and then click OK to close the dialog box.
 - Value Name: command
 - Value Type: UserDefined
 - Value: metadata

- In the IPC Request Data Dialog box, configure the Value Name (the Key Name configured in QM Administrator), the Value Type (selected from the drop-down list) and the Value (the CAD enterprise data chosen from the drop-down list) you want to pass to the QM Recording API. [Figure 6](#) is an example of a completed dialog box for the “balance” metadata field.

Figure 6. IPC Request Data Dialog box for metadata

- Repeat adding more pieces of data to pass from CAD to QM as desired, up to a total of 10 fields. When you are done, the IPC Action Setup dialog box will resemble [Figure 7](#).

Figure 7. Example of IPC Action set up to pass metadata from CAD to the QM Recording API

Name	Value	Value Type
command	metadata	UserDefined
custID	[ENTERPRISE FIELD:Account Number]	DataField
balance	[ENTERPRISE FIELD:Collected Digits]	DataField

- Click OK to save the IPC Action Setup.

Introduction

The QM Server API enables users to search, export, edit, and delete QM call data. It is a Simple Object Access Protocol (SOAP)-based XML web service, and conforms to SOAP 1.2 and Web Services Description Language (WSDL) 1.1.

The QM Server API uses Apache CXF 2.1 to process and handle SOAP messages.

Data Model

SOAP Binding

The QM Server API SOAP bindings for both request and response messages conform to binding specifications outlined in WSDL 1.1, Section 2.5 (“Bindings”).

Character Encoding

All messages are encoded using UTF-8.

Binding Style

While the QM Server API is a remote procedure call (RPC)-style web service, it uses the document/literal SOAP binding style.

Transport Protocols

SOAP allows a number of different transport protocols to carry SOAP messages. The QM Server API uses the HTTP transport protocol with the POST verb. Confidential transport is required, so SSL is used.

SOAP Action Header

SOAP messages sent to the QM Server API should not include the SOAP action HTTP header, since it is not read by the QM Server API.

Port

A SoapPort is an instantiation of a SoapBinding with a specific network address. Since the transport protocol used by the QM Server API is HTTP, that address is a URL. The QM Server API WSDL file on the server that hosts the QM Base services specifies the request URL for all QM Server API operations. See ["Downloading WSDL and XML Schema Files" on page 19](#) for more information.

SOAP Header

Clients must include a SOAP Header that contains authentication information (see ["Authentication" on page 20](#) for more details). Response messages from the QM Server API will not include a SOAP header.

Request and Response Messages

In the QM Server API, each request or response message body consists of a single element from the QM Server API schema. The particular elements for each operation are defined in ["API Operations" on page 21](#).

When a requested operation call is completed, a response message is sent. This response contains the results of the operation.

Fault Messages

When a requested operation call cannot be completed, a SOAP fault message is sent with details of the failure.

Namespaces

The QM Server API uses the following XML namespaces:

<http://www.w3.org/2001/XMLSchema>

<http://www.w3.org/2003/05/soap-envelope>

<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd>

<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd>

<http://www.calabrio.com/qm/serverapi/schema>

Downloading WSDL and XML Schema Files

The WSDL file for the QM Server API can be downloaded from either of these locations:

<http://<server>:8088/serverapi/api?wsdl>

<https://<server>:8448/serverapi/api?wsdl>

The XML schema file for the types used in the QM Server API WSDL file can be downloaded from either of these locations:

<http://<server>:8088/serverapi/api?xsd=api-schema.xsd>

<https://<server>:8448/serverapi/api?xsd=api-schema.xsd>

NOTE: <server> in these URLs is the IP address of the server that hosts the QM Base services.

Authentication

All QM Server API operations require authentication with the user ID and password of a user configured in the QM system.

Including Authentication Information

Authentication information is included in a WS-SecurityUserToken element in the request message's SOAP header.

Example of a SOAP Header

```
<soap:Header>
  <wsse:Security
    xmlns:wsse=
      "http://docs.oasis-open.org/wss/2004/01/
        oasis-200401-wss-wssecurity-secext-1.0.xsd"
    soap:mustUnderstand="true">
    <wsse:UsernameToken
      xmlns:wsu=
        "http://docs.oasis-open.org/wss/2004/01/
          oasis-200401-wss-wssecurity-utility-1.0.xsd"
      wsu:Id="UsernameToken-8454539">
      <wsse:Username>myUserName</wsse:Username>
      <wsse:Password
        Type=
          "http://docs.oasis-open.org/wss/2004/01/
            oasis-200401-wss-username-token-profile-1.0#PasswordText"
        >mySuperSecretPassword</wsse:Password>
      </wsse:UsernameToken>
    </wsse:Security>
  </soap:Header>
```

User Scope

The set of contacts within a given user's scope in the QM Server API is the same as the total set of contacts available to that user in QM Desktop.

The highest role that a user has is the role that will be used with the QM Server API. Any contact the user can see in QM Desktop while logged in with that role is within that user's scope when using the QM Server API. Note, however, that the QM Server API cannot filter for specific groups or teams when used with the manager and supervisor roles.

In order from highest to lowest access, roles that can use the QM Server API are:

- Archive user
- Manager
- Supervisor

API Operations

Search Operation

The search operation searches for contacts in the user's scope that match a set of criteria. It requires authentication.

Request

The request message has one part, a search element as defined in the QM Server API XML schema. The search element can include zero or more of the allowed child elements. Only the metadata element can appear more than once, and only unencrypted metadata can be searched. Only contacts that match all of the included criteria will be returned.

Request Examples

These examples show only the contents of the SOAP body.

Example 1

```
<search xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ANI>1234</ANI>
  <after>2008-02-11T00:00:00+00:00</after>
  <before>2008-03-29T00:00:00+00:00</before>
</search>
```

Example 2

```
<search xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <agentId>agent4</agentId>
  <metadata>
    <key>customerid</key>
    <data>42</data>
  </metadata>
  <metadata>
    <key>calltype</key>
    <data>6</data>
  </metadata>
</search>
```

Response

The response to a search request is a searchResponse element as defined in the QM Server API XML schema. That element contains zero or more customer contact recording (ccr) elements and zero or one truncationBoundaryId elements (see ["Paging" on page 22](#)).

Response Example

This example shows only the contents of the SOAP body.

```
<searchResponse xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ccr>
    <id>4</id>
```

```
<ANI>1234</ANI>
<DNIS>5678</DNIS>
<acdCallId>17356023</acdCallId>
<lastName>Leo</lastName>
<firstName>Fibonacci</firstName>
<agentId>5000.112358</agentId>
<username>doejo</username>
<startTime>2008-05-01T20:04:01+00:00</startTime>
<hasAudio>true</hasAudio>
<hasScreen>false</hasScreen>
</ccr>
<ccr>
  <id>5</id>
  <ANI>1234</ANI>
  <DNIS>9012</DNIS>
  <acdCallId>17356024</acdCallId>
  <lastName>Lucas</lastName>
  <firstName>Francois</firstName>
  <agentId>5000.21347</agentId>
  <username>doejo</username>
  <startTime>2008-05-01T20:04:01+00:00</startTime>
  <hasAudio>true</hasAudio>
  <hasScreen>false</hasScreen>
</ccr>
</searchResponse>
```

Paging

Search results in the response are truncated if they exceed a certain size (the page size). When this happens, the searchResponse element will contain a truncationBoundaryId element specifying the last ccr id in the returned results. If there is no truncationBoundaryId element in a response, either there are no more results, or the full result set has been returned. The next page in the result set can be retrieved by making another search request including this truncationBoundaryId as one of the search criteria.

Export Recording Operation

The exportRecording operation can be used to export the audio or audio and video components of any contact recording in the user's scope. It requires authentication.

Request

The request message has one part, an exportRecording element as defined in the QM Server API XML schema. This element contains two child elements: a ccrId element specifying the contact to retrieve, and a mediatype element specifying the file format to export to. This format must be compatible with the given export type.

Request Examples

These examples show only the contents of the SOAP body.

Example 1

```
<exportRecording xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ccrid>112515</ccrid>
  <mediatype>WMA</mediatype>
</exportRecording>
```

Example 2

```
<exportRecording xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ccrid>112515</ccrid>
  <mediatype>WMV</mediatype>
</exportRecording>
```

Response

The response to an export recording request is an exportRecordingResponse element, as defined in the QM Server API XML schema. This element has one child, an ID element containing the randomly-generated ID of the exported recording. Once the recording has finished exporting, it can be retrieved.

Response Example

This example shows only the contents of the SOAP body.

```
<exportRecordingResponse
  xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <id>431765243</id>
</exportRecordingResponse>
```

Downloading the Exported Recording

An exported recording is available to download for at least one hour after it is finished exporting (see ["Export Progress Operation" on page 23](#)). An HTTP request using the GET verb is used to retrieve the recording by accessing the following URL:

```
http://<server>:8088/serverapi/retrieve?id=<id>
```

The exported recording is immediately deleted after a single access.

Export Progress Operation

The exportProgressRequest operation can be used to request the progress of an export that is in progress or that has been completed but not retrieved. It requires authentication.

Request

The request message has one part, an exportProgressRequest element as defined in the QM Server API XML schema. This IDelement contains one or more child ID elements specifying the exports we are interested in. The ID of an export is the value returned in the response to the exportRecording request.

Request Example

This example shows only the contents of the SOAP body.

```
<exportProgressRequest
  xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <id>431765243</id>
</exportProgressRequest>
```

Response

The response to an export progress request is an exportProgressResponse element as defined in the QM Server API XML schema. This element has one or more child progress elements, each one corresponding to an ID element in the request. Each progress element has two or three child elements:

- An ID element giving the export ID this progress element describes
- A finished element specifying whether the export has finished, and possibly
- An error element describing an error that prevented the recording from successfully exporting, or if no such export id exists. The finished element will always be false if there is an error element.

Response Example

This example shows only the contents of the SOAP body.

```
<exportProgressResponse
  xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <progress>
    <id>431765243</id>
    <finished>>false</finished>
  </progress>
</exportProgressResponse>
```

Delete Recording Operation

The deleteRecording operation can be used to delete the audio and video file of any contact in the user's scope. It requires authentication. Only contacts where all recording elements (audio and video, if it exists) have been uploaded can be deleted using the QM Server API. The deletion mechanism of the QM Server API behaves exactly the same as the mechanism used by DB Cleaner.

Request

The request message has one part, a deleteRecording element as defined in the QM Server API XML schema. This element has a ccrId element as a child element to specify the id of the contact to delete.

Request Example

This example shows only the contents of the SOAP body.

```
<deleteRecording xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ccrId>11235</ccrId>
```



```
</deleteRecording>
```

Response

The response to a delete recording request is a searchResponse element as defined in the QM Server API XML schema. That element has a fileDeleted element as its only child element, which specifies whether a file was deleted. If false, it means that the file was either already deleted or never existed at all.

Response Example

This example shows only the contents of the SOAP body.

```
<deleteRecordingResponse
  xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <fileDeleted>true</fileDeleted>
</deleteRecordingResponse>
```

Edit Metadata Operation

The editMetadata operation can be used to edit existing metadata, add new metadata, or delete metadata for any contact in the user's scope.

Request

The request message has one part, an editMetadata element as defined in the QM Server API XML schema. It contains one ccrId element and zero or more metadata elements. To edit or add data for a given metadata field, include a metadata element with appropriate key and data elements. To remove a given metadata field from a contact, use a metadata element with an empty data element.

Request Example

This example shows only the contents of the SOAP body. Note that the data in the orderDate field is improperly formatted.

```
<editMetadata xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <ccrid>11</ccrid>
  <metadata>
    <key>customer</key>
    <data>24816</data>
  </metadata>
  <metadata>
    <key>orderDate</key>
    <data>one week before yesterday</data>
  </metadata>
  <metadata>
    <key>dne</key>
    <data>some data</data>
  </metadata>
</editMetadata>
```

Response

The response to an edit metadata request is an editMetadataResponse element as defined in the QM Server API XML schema. This element contains one metadataState element for each metadata element in the request. Each metadataState element gives the result of the requested change. The result can be one of the following: success, fieldDoesNotExist, badFormat, or error. Only in the case that the state is success has the data for a particular field been changed.

Response Example

This example shows only the contents of the SOAP body.

```
<editMetadataResponse
  xmlns="http://www.calabrio.com/qm/serverapi/schema">
  <metadataState>
    <key>customer</key>
    <state>success</state>
  </metadataState>
  <metadataState>
    <key>orderDate</key>
    <state>badFormat</state>
  </metadataState>
  <metadataState>
    <key>dne</key>
    <state>fieldDoesNotExist</state>
  </metadataState>
</editMetadataResponse>
```