Cisco Unified Communications Manager XML Developers Guide

Release 8.6(1)

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

Text Part Number: OL-24704-01

Cisco Unified Communications Manager XML Developers Guide, Release 8.6(1)

CONTENTS

INDEX

Preface

This preface includes the following sections:

Purpose

This document describes the following Cisco Unified Communications Manager (Unified CM) (formerly Cisco Unified CallManager) APIs:

- Unified CM AXL implementation allows applications to modify the Unified CM system database. Be aware that AXL is not intended as a real-time API but as a provisioning and configuration API.
- Unified CM real-time information, performance counters, and database information exposure occur through the Serviceability XML API.
- Unified CM Extension Mobility Service provides a rich API, which enables extension mobility on Cisco Unified IP phones and allows application control over authentication, scheduling, and availability. It allows a device, usually a Cisco Unified IP Phone, to temporarily embody a new device profile, including lines, speed dials, and services. An application that uses the Cisco Unified CM Mobility Service represents an IP phone service that allows a user to log in by entering a userID and PIN. The architecture and implementation of the Cisco Unified CM Extension Mobility Service make many other applications possible.

  Examples include:

  - An application that automatically activates phones for employees when they reserve a particular desk for a particular time (the scheduling application)
  - A lobby phone does not have a line appearance until a user logs in

- Unified CM Web Dialer application, which is installed on a Unified CM server, enables click-to-dial functionality by creating hyperlinked telephone numbers in a company directory. This functionality allows users to make calls from a web page by clicking the telephone number of the person that they are trying to call. The Web Dialer application, which has a SOAP interface, uses JavaScript to provide the web page functionality.
- Cisco Unified Routing Rules XML interface supports call-routing decisions for Cisco Unified Communications Manager. Cisco Unified Communication Manager 8.0(1) supports the external call control (ECC) feature, which enables an adjunct Route Server to make call-routing decisions for Cisco Unified Communications Manager by using the 8.0(1) Cisco Unified Routing Rules Interface.

Audience

The *Cisco Unified Communication Manager Developers Guide* provides information for developers who write applications that extend the functionality of the APIs that are described in this document.

This guide assumes the developer has knowledge of a high-level programming language such as C++, Java, or an equivalent language. You must also have knowledge or experience in the following areas:

- Extensible Markup Language (XML)
- Hypertext Markup Language (HTML)
- Hypertext Transport Protocol (HTTP)
- Simple Object Access Protocol (SOAP) 1.1
- Socket programming
- TCP/IP Protocol
- Web Service Definition Language (WSDL) 1.1
- Secure Sockets Layer (SSL)

In addition, users of the Unified CM APIs must have a firm grasp of XML Schema. For more information about XML Schema, refer to http://www.w3.org/TR/xmlschema-0/.

The developer must also have an understanding of Unified CM and its applications. The "Related Documentation" section on page xiv lists documents for Unified CM and other related technologies.

Organization

This document is organized as follows:

Part	Chapter	Description
	Chapter 1, "Overview"	Describes the Cisco Unified Communications Manager interfaces.
Part 1 Administrative XML	Chapter 2, "Administrative XML Programming"	Describes the Administrative XML Layer (AXL) API, which provides a mechanism for inserting, retrieving, updating, and removing data from the database by using an XML SOAP interface. This API lets you access Unified CM data by using XML and receive the data in XML form.
	Chapter 3, "Administrative XML Operations by Release"	Lists new, changed, and deprecated Administrative XML (AXL) operations by release.
Part 2 Serviceability XML	Chapter 4, "Serviceability XML Programming"	Describes the Serviceability XML APIs. Unified CM real-time information, performance counters, and database information exposure occurs through the Serviceability XML APIs.
	Chapter 5, "Serviceability XML Operations by Release"	Lists new, changed, and deprecated serviceability XML operations by release.
Part 3 Extension Mobility Service API	Chapter 6, "Cisco Extension Mobility Service API"	Describes high-level concepts that are important in understanding the Cisco Extension Mobility Service and provides an overview of configuring EM services, messages, message DTDs, and error codes.
	Chapter 7, "Cisco Extension Mobility Operations By Release"	Lists new, changed, and deprecated Extension Mobility Operations by release.
Part 4 Web Dialer API	Chapter 8, "Cisco Web Dialer API Programming"	Describes the Simple Object Access Protocol (SOAP) and HTML over HTTP (and HTTPS) interfaces that are used to develop JavaScript-based directory search web pages and applications for Cisco Web Dialer.
	Chapter 9, "Cisco Web Dialer Operations By Release"	Lists new, changed, and deprecated Web Dialer operations by release.
Part 5 Routing Rules API	Chapter 10, "Cisco Unified Routing Rules Interface"	Describes the Cisco Unified Policy XML interface. It explains the Access Control Markup Language (XACML) call routing requests and response, and the Call Instruction XML (CIXML) obligations used in external call control.

Related Documentation

This section lists documents and URLs that provide information on Unified CM, Cisco Unified IP Phones, and the technologies that are required to develop applications.

Release Notes for Cisco Unified Communications Manager, Release x.x.x	For release notes for various releases of Cisco Unified Communication Manager.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_release_notes_list.html*
Cisco Unified Communications Manager Admin XML Interface Spec	For diagrams, name spaces, attributes, code examples, and inter-relationships of all methods and handlers.	*http://developer.cisco.com/web/axl/docs*
Cisco Unified Communications Manager Data Dictionary	For information on the data stored in the primary Cisco Unified Communications Manager database.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_reference_guides_list.html*
Cisco Unified Communications Manager Administration Guide	For step-by-step instructions for configuring, maintaining, and administering the Cisco Unified Communications Manager voice over IP network.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_maintenance_guides_list.html*
Cisco Unified Communications Manager System Guide	For conceptual information about Cisco Unified Communications Manager and its components, as well as tips for setting up features by using Cisco Unified Communications Manager Administration.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_maintenance_guides_list.html*
Cisco Unified Communications Manager Features and Services Guide	For information to understand, install, configure, manage, and use Cisco Unified Communications Manager features.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_maintenance_guides_list.html*
Cisco Unified Serviceability Administration Guide	For information on serviceability alarms, traces, tools, reports, and SNMP.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_maintenance_guides_list.html*
Cisco Unified IP Phones and Services	A suite of documents that relate to the installation and configuration of Cisco Unified IP Phones	*http://www.cisco.com/cisco/web/psa/default.html* Select **Voice and Unified Communications>IP Telephony>IP Phones**
Cisco Unified Communications Manager Documentation Guide	For documentation updates in specific releases of Cisco Unified Communications Manager.	*http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_documentation_roadmaps_list.html*

Cisco Developer Community Tech Centers	*Administration XML (AXL) Tech Center*—for forums, blogs, wikis, and documentation on AXL.	*http://developer.cisco.com/web/axl/home*
	Serviceability XML Tech Center—for forums, blogs, wikis, and documentation on Serviceability XML.	*http://developer.cisco.com/web/sxml/home*
	Extension Mobility Service Tech Center—or forums, blogs, wikis, and documentation on Extension Mobility APIs.	*http://developer.cisco.com/web/emapi/home*
	WebDialer Tech Center—or forums, blogs, wikis, and documentation on WebDialer.	*http://developer.cisco.com/web/webdialer*
Simple Object Access Protocol (SOAP) 1.1	For latest SOAP versions and notes.	*http://www.w3.org/TR/SOAP/*
Web Service Definition Language (WSDL) 1.1	For more information on Web Services Description Language (WSDL) 1.1	*http://www.w3.org/TR/wsdl*
SOAP Tutorial	For tutorial on Simple Object Access Protocol.	*http://www.w3schools.com/soap/default.asp*
WSDL Tutorial	For tutorial on Web Services Description Language.	*http://www.w3schools.com/wsdl/default.asp*
SoapAgent.com	For open SOAP directory with links to articles, tutorials, and white papers.	*http://www.soapagent.com/*

Cisco Developer Network

The Cisco Developer Network (CDN) portal provides access to multiple Cisco technology developer interfaces and collaborative support communities. CDN also provides formalized support services for these interfaces to enable developers, customers, and partners to accelerate their development. The formalized process provides access to CDN Engineers who are an extension of the product technology engineering teams. CDN Engineers have access to the resources necessary to provide expert support in a timely manner.

The Cisco Developer Network Program is designed for businesses (IHV's and ISV's) interested in going to market with Cisco. The CDN Program enables members to develop compelling solutions that unify data, voice, video, and mobile communications on Cisco's powerful communications platform. The program also allows members to take advantage of Cisco's brand, market leadership position, and installed base to help drive positive business results for themselves and their customers.

For additional information about the CDN Program and CDN support services go to *http://developer.cisco.com/web/devservices*

Conventions

This document uses the following conventions:

Convention	Description
boldface font	Commands and keywords are in **boldface**.
italic font	Arguments for which you supply values are in *italics*.
[ ]	Elements in square brackets are optional.
{ x | y | z }	Alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A non-quoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`screen` font	Terminal sessions and information the system displays are in `screen` font.
`boldface screen` font	Information you must enter is in **`boldface screen`** font.
`italic screen` font	Arguments for which you supply values are in *`italic screen`* font.
^	The symbol ^ represents the key labeled Control—for example, the key combination ^D in a screen display means hold down the Control key while you press the D key.
< >	Non-printing characters, such as passwords, are in angle brackets.

Notes use the following conventions:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Timesavers use the following conventions:

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Tips use the following conventions:

Tip Means *the following are useful tips.*

Cautions use the following conventions:

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Warnings use the following conventions:

Warning **This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, you must be aware of the hazards involved with electrical circuitry and familiar with standard practices for preventing accidents.**

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

Cisco Product Security Overview

This product contains cryptographic features and is subject to United States and local country laws governing import, export, transfer and use. Delivery of Cisco cryptographic products does not imply third-party authority to import, export, distribute or use encryption. Importers, exporters, distributors and users are responsible for compliance with U.S. and local country laws. By using this product you agree to comply with applicable laws and regulations. If you are unable to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at: http://www.cisco.com/wwl/export/crypto/tool/stqrg.html. If you require further assistance please contact us by sending e-mail to export@cisco.com.

CHAPTER 1

Overview

Cisco Unified Communications Manager (Unified CM) is the powerful call-processing component of the Cisco Unified Communications Solution. It is a scalable, distributable, and highly available enterprise IP telephony call-processing solution. Unified CM acts as the platform for collaborative communication and as such supports a wide array of features. In order to provision, invoke the features, monitor, and control such a powerful system, Unified CM supports different interface types.

This chapter gives an introduction to the different interfaces of Unified CM and includes the following sections:

Cisco Unified Communications Manager Interfaces

The interface types supported by Unified CM are divided into the following types:

Provisioning Interfaces

The following are the provisioning interfaces of Unified CM:

- Administration XML
- Cisco Extension Mobility service

Administrative XML

The Administration XML (AXL) API provides a mechanism for inserting, retrieving, updating and removing data from the Unified CM configuration database using an eXtensible Markup Language (XML) Simple Object Access Protocol (SOAP) interface. This allows a programmer to access Unified CM provisioning services using XML and exchange data in XML form, instead of using a binary library or DLL. The AXL methods, referred to as requests, are performed using a combination of HTTP and

SOAP. SOAP is an XML remote procedure call protocol. Users perform requests by sending XML data to the Unified CM Publisher server. The publisher then returns the AXL response, which is also a SOAP message.

This guide gives detailed information on the AXL API see Part 1, Administrative XML.

For more information, see the Administrative XML Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/axl/home

Cisco Extension Mobility

The Cisco Extension Mobility (Extension Mobility) service, a feature of Unified CM, allows a device, usually a Cisco Unified IP Phone, to temporarily embody a new device profile, including lines, speed dials, and services. It enables users to temporarily access their individual Cisco Unified IP Phone configuration, such as their line appearances, services, and speed dials, from other Cisco Unified IP Phones. The Extension Mobility service works by downloading a new configuration file to the phone. Unified CM dynamically generates this new configuration file based on information about the user who is logging in. You can use the XML-based Extension Mobility service API with your applications, so they can take advantage of Extension Mobility service functionality.

This guide gives detailed information on the Extension Mobility APIs, see Part 3, Extension Mobility Service API.

For more information, see the Extension Mobility API Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/emapi/home

Device Monitoring and Call Control Interfaces

The following are the device monitoring and call control interfaces of Unified CM:

- Cisco TAPI and Media Driver
- Cisco JTAPI
- Cisco Web Dialer

Cisco TAPI and Media Driver

Unified CM exposes sophisticated call control of IP telephony devices and soft-clients via the Computer Telephony TAPI interface. Cisco's Telephone Service Provider (TSP) and Media Driver interface enables custom applications to monitor telephony-enabled devices and call events, establish first- and third-party call control, and interact with the media layer to terminate media, play announcements, record calls.

Information on Cisco TAPI and Media Driver is beyond the scope of this guide. For information of Cisco TAPI and Media Driver, see *Cisco Unified TAPI Developers Guide for Cisco Unified Communications Manager* for relevant release of Unified CM at the following location:

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

For more information, see the TAPI and Media Driver Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/tapi/home

Cisco JTAPI

Unified CM exposes sophisticated call control of IP telephony devices and soft-clients via the Computer Telephony JTAPI interface. Cisco's JTAPI enables custom applications to monitor telephony-enabled devices and call events, as well as establish first- and third-party call control.

Information on Cisco JTAPI is beyond the scope of this guide. For information on Cisco JTAPI, see *Cisco Unified JTAPI Developers Guide for Cisco Unified Communications Manager* for relevant release of Unified CM at the following location:

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

For more information, see the JTAPI Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/jtapi/home

Cisco Web Dialer

The Web Dialer, which is installed on a Unified CM server, allows Cisco Unified IP Phone users to make calls from web and desktop applications. For example, the Web Dialer uses hyper linked telephone numbers in a company directory to allow users to make calls from a web page by clicking the telephone number of the person that they are trying to call. The two main components of Web Dialer comprise the Web Dialer Servlet and the Redirector Servlet.

This guide gives detailed information on the Web Dialer API, see Part 4, Web Dialer API.

For more information, see the Web Dialer Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/wd/home

Serviceability Interfaces

The following are the serviceability interfaces of Unified CM:

- Serviceability XML
- SNMP/MIBs

Serviceability XML

A collection of services and tools designed to monitor, diagnose, and address issues specific to Unified CM. Serviceabiltiy XML interface:

- Provides platform, service and application performance counters to monitor the health of Unified CM hardware and software
- Provides real-time device and CTI connection status to monitor the health of phones, devices, and applications connected to Unified CM.
- Enables remote control (Start/Stop/Restart) of Unified CM services.
- Collects and packages Unified CM trace files and logs for troubleshooting and analysis.
- Provides applications with Call Detail Record files based on search criteria.
- Provides management consoles with SNMP data specific to Unified CM hardware and software.

This guide gives detailed information on the Serviceability XML APIs, see Part 2, Serviceability XML.

For more information, see the Serviceability XML Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/sxml/home

SNMP/MIBs

SNMP interface allows external applications to query and report various Unified CM entities. It provides information on the connectivity of the Unified Communication Manager to other devices in the network, including syslog information.

The MIBs supported by Unified CM includes:

- Cisco-CCM-MIB, CISCO-CDP-MIB, Cisco-syslog-MIB
- Standard MIBs like MIB II, SYSAPPL-MIB, HOST RESOURCES-MIB
- Vendor MIBs

For more information, see the SNMP/MIB Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/sxml/home

Routing Rules Interface

Cisco Unified Communication Manager 8.0(1) supports the external call control (ECC) feature, which enables an adjunct route server to make call-routing decisions for Cisco Unified Communications Manager by using the 8.0(1) Cisco Unified Routing Rules Interface. When you configure external call control, Cisco Unified Communications Manager issues a route request that contains the calling party and called party information to the adjunct route server. The adjunct route server receives the request, applies appropriate business logic, and returns a route response that instructs Cisco Unified Communications Manager on how the call should get routed, along with any additional call treatment that should be applied.

For more information, see the Routing Rules Interface Tech Center on the Cisco Developer Network at http://developer.cisco.com/web/curri/home

Development Guidelines

Cisco maintains a policy of interface backward compatibility for at least one previous major release of Cisco Unified Communications Manager (Cisco Unified CM). Cisco still requires Cisco Technology Developer Program member applications to be retested and updated as necessary to maintain compatibility with each new major release of Cisco Unified CM.

The following practices are recommended to all developers, including those in the Cisco Technology Developer Program, to reduce the number and extent of any updates that may be necessary:

- The order of events and/or messages may change. Developers should not depend on the order of events or messages. For example, where a feature invocation involves two or more independent transactions, the events or messages may be interleaved. Events related to the second transaction may precede messages related to the first. Additionally, events or messages can be delayed due to situations beyond control of the interface (for example, network or transport failures). Applications should be able to recover from out of order events or messages, even when the order is required for protocol operation.
- The order of elements within the interface event or message may change, within the constraints of the protocol specification. Developers must avoid unnecessary dependence on the order of elements to interpret information.

- New interface events, methods, responses, headers, parameters, attributes, other elements, or new values of existing elements, may be introduced. Developers must disregard or provide generic treatments where necessary for any unknown elements or unknown values of known elements encountered.
- Previous interface events, methods, responses, headers, parameters, attributes, and other elements, will remain, and will maintain their previous meaning and behavior to the extent possible and consistent with the need to correct defects.
- Applications must not be dependent on interface behavior resulting from defects (behavior not consistent with published interface specifications) since the behavior can change when defect is fixed.
- Use of deprecated methods, handlers, events, responses, headers, parameters, attributes, or other elements must be removed from applications as soon as possible to avoid issues when those deprecated items are removed from Cisco Unified CM.
- Application Developers must be aware that not all new features and new supported devices (for example, phones) will be forward compatible. New features and devices may require application modifications to be compatible and/or to make use of the new features/devices.

What is New in Cisco Unified Communications Manager 8.6(1) API

This section gives information on the changes in the following APIs in the Unified CM release 8.6(1):

- Administrative XML API, page 1-5
- Serviceability XML API, page 1-6
- Extension Mobility API, page 1-6
- Web Dialer API, page 1-6
- Routing Rules API, page 1-6

Note Information on changes in Cisco JTAPI and Cisco TAPI and Media Driver API are beyond the scope of this guide.
For information of Cisco TAPI and Media Driver, see *Cisco Unified TAPI Developers Guide for Cisco Unified Communications Manager* and for information on Cisco JTAPI, see *Cisco Unified JTAPI Developers Guide for Cisco Unified Communications Manager* for relevant release of Unified CM at the following location:
http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_reference_guides_list.html

Administrative XML API

The following are the changes in the Administrative XML API in Unified CM release 8.6(1).

- A new AXL schema is provided in the first major (X.0) and first minor (X.5) release of each Unified CM train. As such, Unified CM 8.6(1) does not contain a new AXL schema. Developers should use the AXL 8.5 schema to configure features and device settings which were introduced (or available) in the Unified CM 8.5(x) release. New Unified CM 8.6 devices can be configured using the AXL 8.5 schema.

Developers should use the direct sql methods ExecuteSQLQuery and ExecuteSQLUpdate to configure new features and device settings added in Unified CM 8.6. Unified CM 9.0(1) will include a new AXL 9.0(1) schema which will include all 8.6(x) database objects.

For more information on the changes in the Administrative XML API, see the New and Changed Information in Unified CM Release 8.6(1) section in the Chapter 2, "Administrative XML Programming".

Serviceability XML API

The following change was made to the Serviceability XML API in Unified CM release 8.6(1).

- SelectCmDevice API returns device information for a maximum of 1000 devices instead of the earlier 200 devices.

For more information, see the section MaxReturnedDevices, page 4-31.

Extension Mobility API

There are no changes in the Extension Mobility API in Unified CM release 8.6(1).

Web Dialer API

There are no changes in the Web Dialer API in Unified CM release 8.6(1).

Routing Rules API

There are no changes in the Routing Rules API in Unified CM release 8.6(1).

PART 1

Administrative XML

CHAPTER 2

Administrative XML Programming

This chapter describes the Administrative XML Layer (AXL) Application Programming Interface (API). It contains the following sections:

Overview

The AXL API provides a mechanism for inserting, retrieving, updating, and removing data from the Unified CM database by using an eXtensible Markup Language (XML) Simple Object Access Protocol (SOAP) interface. This approach allows a programmer to access the database by using XML and receive the data in XML form, instead of by using a binary library or DLL.

The AXL API methods, known as requests, use a combination of HTTPS and SOAP. SOAP is an XML remote procedure call (RPC) protocol. The server receives the XML structures and executes the request. If the request completes successfully, the system returns the appropriate AXL response. All responses are named identically to the associated requests, except that the word "Response" is appended.

For example, the XML response that is returned from an addPhone request is named addPhoneResponse.

If an error occurs, an XML error structure is returned, wrapped inside a SOAP Fault structure (see the "AXL Error Codes" section on page 2-163).

The AXL-SOAP web service is disabled by default on all Cisco Unified Communications Manager (Unified CM) servers that are running version 5.x or later. You should start the service before using the AXL APIs.

To access all AXL SOAP API downloads and AXL requests and responses that are found in this chapter, refer to http://www.cisco.com/pcgi-bin/dev_support/access_level/product_support.

This chapter assumes that the developer has knowledge of a high-level programming language such as C++, Java, or an equivalent language, and has knowledge of SOAP.

Developers must also have knowledge or experience in the following areas:

- TCP/IP Protocol
- Hypertext Transport Protocol (specifically HTTPS)
- Socket programming
- XML

Users of the AXL API must have a firm grasp of XML syntax and Schema, which is used to define the AXL requests, responses, and errors. For more information about XML Schema, refer to http://www.w3.org/TR/xmlschema-0/. For more information about XML syntax/grammar, refer to http://www.w3.org/TR/rdf-syntax-grammar/.

Caution The AXL API allows you to modify the Unified CM system database. Use caution when using AXL because each API call affects the system. Misuse of the API can lead to dropped calls and slower performance. AXL should act as a provisioning and configuration API, not as a real-time API.

AXL Compliance

The Unified CM AXL implementation complies with XML Schema 1.0, which was tested for XML Schema compliance with a third-party application that is called XML Spy version 4.x. Early versions of the MSXML schema validator did not support enough of the XML Schema 1.0 recommendation to be used.

The Unified CM AXL implementation also complies with SOAP 1.1 as defined by the World Wide Web Consortium as well as HTTPS 1.1. The AXL API runs as an independent service that can be accessed only via HTTPS.

New and Changed Information

The following sections provide information on the changes in the AXL APIs in Unified CM release 8.6(1) and the previous releases:

For information on the new, changed, or removed AXL API methods from the interface library, see the Chapter 3, "Administrative XML Operations by Release".

New and Changed Information in Unified CM Release 8.6(1)

The AXL APIs in Unified CM Release 8.6(1) are compatible with all previous releases of Unified CM. For additional details about the Unified CM database schema changes, see the *Unified CM Data Dictionary for Release 8.6(1)*.

The following sections describe API changes in Unified CM Release 8.6(1):

- A new AXL schema is provided in the first major (X.0) and first minor (X.5) release of each Unified CM train. As such, Unified CM 8.6(1) does not contain a new AXL schema. Developers should use the AXL 8.5 schema to configure features and device settings which were introduced (or available) in the Unified CM 8.5(x) release. New Unified CM 8.6 devices can be configured using the AXL 8.5 schema. Developers should use the direct SQL methods ExecuteSQLQuery and ExecuteSQLUpdate to configure new features and device settings added in Unified CM 8.6. Unified CM 9.0(1) will include a new AXL 9.0(1) schema which will include all 8.6(x) database objects.

Note If you are using the U.S. export unrestricted version of Cisco Unified CM 8.6, all API operations of VpnGateway, VpnGroup, and VpnProfile will fail because the corresponding DB tables are denied permission. Also, vpnGroupName and vpnProfileName of the CommonPhoneConfig DB table are denied permission in a U.S. export unrestricted version. Hence, all CommonPhoneConfig API operations will fail when they are executed with these tags.

When the backward-compatible versions (which support VPN) 8.0 and 8.5 AXL schemas are used for performing AXL operation on the U.S. export unrestricted Cisco Unified CM 8.6 version, they will also show the same behavior.

New and Changed Information in Previous Releases of Unified CM

The following sections provide the new and changed information in the older releases of Unified CM:

For information about new, changed, or deprecated AXL API methods from the interface library, see Chapter 3, "Administrative XML Operations by Release".

New Information for Unified CM 8.5(1)

The AXL APIs in Unified CM 8.5(1) APIs are compatible with all previous releases of Unified CM. For additional details about the Unified CM database schema changes, see the *Unified CM Data Dictionary for Release 8.5(1)*.

The following sections describe API updates in Unified CM 8.5(1):

Note If you are using 8.5 AXL schemas to send AXL requests to the U.S. export unrestricted version of Cisco Unified CM 8.6, all API operations of VpnGateway, VpnGroup, and VpnProfile will fail. Also, all CommonPhoneConfig API operations will fail if they are executed with vpnGroupName and vpnProfileName tags.

New APIs

Table 2-1 provides information on the new APIs in Unified CM release 8.5(1).

Changed APIs

Table 2-1 provides information on the modified APIs in Unified CM release 8.5(1).

Update to AXL Schema Versioning

The following changes are made in the version support matrix in Unified CM release 8.5(1):

- The 8.5 schema is added.
- The default schema when a SOAPAction header is not specified is now 7.0 (formerly 6.1).

For more information, see the AXL Versioning Support, page 2-134.

Table 2-1 Changed Operations in Unified CM 8.5(1)

API	New tags	Deprecated tags
CommonPhoneConfig	featureControlPolicy	—
DeviceProfile	featureControlPolicy	—
doAuthenticateUser	daysToExpiry	—
FallbackFeatureConfig	clearImeCallDelayTimer dtmfInterDigitDelayTimer postConnectFallbackDelayTimer fallbackSplitDelayTimer	—
H323Trunk	runOnEveryNode destinations - destination - addressIpv4, sortOrder	—
ImeClient	ccmExternalIpMaps - ccmExternalIpMap - callManagerName, ipAddressHost, port	—
Phone	featureControlPolicy	—
RemoteDestination	mobilityProfileName	—
RouteList	runOnEveryNode	—

Table 2-1 Changed Operations in Unified CM 8.5(1)

API	New tags	Deprecated tags
SipProfile	deliverConferenceBridgeIdentifier earlyOfferSupportForVoiceCall enableOutboundOptionsPing optionsPingIntervalWhenStatusNotOK optionsPingIntervalWhenStatusOK sendRecvSDPInMidCallInvite sipOptionsRetryCount sipOptionsRetryTimer	—
SipTrunk	asn1RoseOidEncoding destinations - destination - addressIpv4, addressIpv6, port, sortOrder enableQsigUtf8 pathReplacementSupport qsigVariant runOnEveryNode scriptParameters scriptTraceEnabled sipNormalizationScriptName trunkTrafficSecure tunneledProtocol	destinationAddress destAddrIsSrv destinationport destinationAddressIpv6 sipTrunkType(unsupported only in update method)

Table 2-2 New Operations in Unified CM 8.5(1)

API	Operation	Added Tags
EnterpriseFeatureAccessCofiguration	addEnterpriseFeatureAccessCofiguration getEnterpriseFeatureAccessCofiguration listEnterpriseFeatureAccessCofiguration removeEnterpriseFeatureAccessCofiguration updateEnterpriseFeatureAccessCofiguration	pattern routePartitionName description isDefaultEafNumber
HandOffConfiguration	addHandOffConfiguration getHandOffConfiguration removeHandOffConfiguration updateHandOffConfiguration	pattern routePartitionName

Table 2-2 New Operations in Unified CM 8.5(1) (continued)

API	Operation	Added Tags
ImeLearnedRoutes	getImeLearnedRoutes removeImeLearnedRoutes updateImeLearnedRoutes	adminEnabled domain e164Did expiresOn learnedOn signaling
LdapSync	doLdapSync	name sync
LdapSyncStatus	getLdapSyncStatus	name uuid
MobilityProfile	addMobilityProfile getMobilityProfile listMobilityProfile removeMobilityProfile updateMobilityProfile	description dirn - pattern dvofServiceAccessNumber dvorCallerId mobileClientCallingOption name routePartitionName
SoftkeySet	getSoftkeySet updateSoftkeySet	name callStates - callState - callStateName,softkeys softkeys - softkey - potitionId, softkeyName
UpdateRemoteCluster	doUpdateRemoteCluster	server clusterId

New Information for Unified CM 8.0(1)

The AXL APIs in Unified CM 8.0(1) APIs are compatible with all previous releases of Unified CM. For additional details about the Unified CM database schema changes, see the *Unified CM Data Dictionary for Release 8.0(1)*.

The following sections describe API updates in Unified CM 8.0(1):

- New APIs, page 2-8
- Changed APIs, page 2-42
- Schema and Other Changes, page 2-102

Note If you are using 8.0 AXL schemas to send AXL requests to the U.S. export unrestricted version of Cisco Unified CM 8.6, all API operations of VpnGateway, VpnGroup, and VpnProfile will fail. Also, all CommonPhoneConfig API operations will fail if they are executed with vpnGroupName and vpnProfileName tags.

New APIs

Table 2-3 describes new operations in Unified CM 8.0(1).

Table 2-3 New Operations in Unified CM 8.0(1)

Operation	Purpose	Added Tags
addSafSecurityProfile updateSafSecurityProfile getSafSecurityProfile removeSafSecurityProfile	To support Call Control Discovery feature	name (mandatory) description userid (mandatory) password (mandatory)
addSafForwarder updateSafForwarder getSafForwarder removeSafForwarder	To support Call Control Discovery feature	name(mandatory) description clientLabel (mandatory) safSecurityProfile(mandatory) ipAddress(mandatory) port safReconnectInterval safNotificationsWindowSize associatedCucms associatedCucm (mandatory) callManagerName (mandatory)
addCcdAdvertisingService updateCcdAdvertisingService getCcdAdvertisingService removeCcdAdvertisingService	To support Call Control Discovery feature	name(mandatory) description isActivated hostDnGroup(mandatory) safSipTrunk(mandatory) safH323Trunk(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addCcdRequestingService updateCcdRequestingService getCcdRequestingService removeCcdRequestingService	To support Call Control Discovery feature	name(mandatory) description isActivated routePartitionName learnedPatternPrefix pstnPrefix associatedTrunks associatedTrunk(mandatory) trunkName (mandatory)
addCcdHostedDN updateCcdHostedDN getCcdHostedDN removeCcdHostedDN	To support Call Control Discovery feature	hostedPattern(mandatory) description CcdHostedDNGroup(mandatory) pstnFailoverStripDigits pstnFailoverPrependDigits usePstnFailover
addCcdHostedDNGroup updateCcdHostedDNGroup getCcdHostedDNGroup removeCcdHostedDNGroup	To support Call Control Discovery feature	name(mandatory) description pstnFailoverStripDigits pstnFailoverPrependDigits usePstnFailover
addSafCcdPurgeBlockLearnedRoutes updateSafCcdPurgeBlockLearnedRoutes getSafCcdPurgeBlockLearnedRoutes removeSafCcdPurgeBlockLearnedRoutes	To support Call Control Discovery feature	learnedPattern learnedPatternPrefix callControlIdentity ipAddress.
updateInterClusterServiceProfile getInterClusterServiceProfile	To support Extension Mobility Cross Clusters feature	interClusterService (mandatory) isActivated sipTrunkName
addRemoteCluster updateRemoteCluster getRemoteCluster removeRemoteCluster	To support Extension Mobility Cross Clusters feature	clusterId(mandatory) description fullyQualifiedName(mandatory) iemcc enabled(mandatory) pstnAccess enabled(mandatory) rsvpAgent

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		enabled(mandatory)
addExternalCallControlProfile updateExternalCallControlProfile getExternalCallControlProfile removeExternalCallControlProfile	To support Call Intercept feature	name (mandatory) primaryUri(mandatory) secondaryURI enableDigestAuthSecondary appUserDigestUserSecondary enableLoadBalancing routingRequestTimer diversionReroutingCssName callTreatmentOnFailure (mandatory)
addCumaServerSecurityProfile updateCumaServerSecurityProfile getCumaServerSecurityProfile removeCumaServerSecurityProfile	To support Mobility SIP Extension feature	name (mandatory) description securityMode transportType(mandatory) x509SubjectName serverIpHostName (mandatory)
addImeClient updateImeClient getImeClient removeImeClient	To support Unified B2B Link feature	name(mandatory) description domain(mandatory) isActivated sipTrunkName(mandatory) primaryImeServerName(mandatory) secondaryImeServerName learnedRouteFilterGroupName exclusionNumberGroupName firewallName members member enrolledPatternGroupName
addImeE164Transformation updateImeE164Transformation getImeE164Transformation removeImeE164Transformation	To support Unified B2B Link feature	name(mandatory) description cgpnTransformationCssName isCgpnPreTransformation cdpnTransformationCssName isCdpnPreTransformation

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		isCdpnPreTransformation incomingCgpnTransformationProfileName incomingCdpnTransformationProfileName
addImeRouteFilterGroup updateImeRouteFilterGroup getImeRouteFilterGroup removeImeRouteFilterGroup	To support Unified B2B Link feature	name(mandatory) description groupTrustSetting
addImeRouteFilterElement updateImeRouteFilterElement getImeRouteFilterElement removeImeRouteFilterElement	To support Unified B2B Link feature	name(mandatory) description elementType imeRouteFilterGroupName(mandatory)
addImeEnrolledPattern updateImeEnrolledPattern getImeEnrolledPattern removeImeEnrolledPattern	To support Unified B2B Link feature	pattern description imeEnrolledPatternGroupName(mandatory)
addImeEnrolledPatternGroup updateImeEnrolledPatternGroup getImeEnrolledPatternGroup removeImeEnrolledPatternGroup	To support Unified B2B Link feature	name(mandatory) description fallbackProfileName isPatternAllAlias
addImeExclusionNumber updateImeExclusionNumber getImeExclusionNumber removeImeExclusionNumber	To support Unified B2B Link feature	pattern(mandatory) description imeExclusionNumberGroupName(mandatory)
addImeExclusionNumberGroup updateImeExclusionNumber getImeExclusionNumber removeImeExclusionNumber	To support Unified B2B Link feature	name(mandatory) description
addImeFirewall updateImeFirewall getImeFirewall removeImeFirewall	To support Unified B2B Link feature	name(mandatory) description ipAddress(mandatory) port

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addImeServer updateImeServer getImeServer removeImeServer	To support Unified B2B Link feature	name(mandatory) description ipAddress(mandatory) port deviceSecurityMode applicationUser(mandatory) reconnectInterval
addFallbackProfile updateFallbackProfile getFallbackProfile removeFallbackProfile	To support Unified B2B Link feature	name(mandatory) description advertisedFallbackDirectoryE164Number QosSensistivityLevel callCss(mandatory) callAnswerTimer directoryNumberPartition directoryNumber numberOfDigitsforCallerIDPartialMatch
addPhoneNTP updatePhoneNTP getPhoneNTP removePhoneNTP	To add, update, get, and remove PhoneNTP	ipAddress description mode(mandatory)
addDateTimeGroup getDateTimeGroup removeDateTimeGroup	To add, get, and remove DateTimeGroup	name(mandatory) timeZone(mandatory) separator(mandatory) dateformat(mandatory) timeformat(mandatory) phoneNtpReferences selectedPhoneNtpReference phoneNtpName(mandatory) selectionOrder(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
updateDateTimeGroup	To update DateTimeGroup	newName timeZone separator dateformat timeFormat removePhoneNtpReferences selectedPhoneNtpReference(mandatory) phoneNtpName(mandatory) selectionOrder(mandatory) addPhoneNtpReferences selectedPhoneNtpReference(mandatory) phoneNtpName(mandatory) selectionOrder(mandatory) phoneNtpReferences selectedPhoneNtpReference(mandatory) phoneNtpName(mandatory) selectionOrder(mandatory)
addPresenceGroup updatePresenceGroup getPresenceGroup removePresenceGroup	To add, update, get, and remove PresenceGroup	name(mandatory) description presenceGroups presenceGroup presenceGroupName(mandatory) subscriptionPermission(mandatory)
addMlppDomain updateMlppDomain getMlppDomain removeMlppDomain	To add, update, get and remove MlppDomain	domainName(mandatory) domainId(mandatory)
addPhoneSecurityProfile updatePhoneSecurityProfile getPhoneSecurityProfile removePhoneSecurityProfile	To add, update, get, and remove PhoneSecurityProfile	phoneType(mandatory) protocol(mandatory) name(mandatory) description deviceSecurityMode authenticationMode keySize tftpEncryptedConfig

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		nonceValidityTime transportType sipPhonePort enableDigestAuthentication excludeDigestCredentials
addApplicationServer getApplicationServer removeApplicationServer	To add, get, and remove ApplicationServer	appServerType(mandatory) name(mandatory) ipAddress appUsers selectedAppUser(mandatory) appUserName(mandatory) content url endUserUrl processNodeName endUsers selectedEndUser(mandatory) endUserName(mandatory) content
updateApplicationServer	To update ApplicationServer	newName ipAddress removeAppUsers selectedAppUser(mandatory) appUserName(mandatory) content addAppUsers selectedAppUser(mandatory) appUserName(mandatory) content appUsers selectedAppUser(mandatory) appUserName(mandatory) content

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addAppDialRules getAppDialRules updateAppDialRules removeAppDialRules	To add, update, get, and remove appDialRules	name (Mandatory) description numberBeginWith numberOfDigits (Mandatory) digitsToBeRemoved (Mandatory) prefixPattern priority (Mandatory)
addDirectoryLookupDialRules getDirectoryLookupDialRules updateDirectoryLookupDialRules removeDirectoryLookupDialRules	To add, update, get, and remove DirectoryLookupDialRules	name (Mandatory) description numberBeginWith numberOfDigits (Mandatory) digitsToBeRemoved (Mandatory) prefixPattern priority (Mandatory)
addSipDialRules getSipDialRules updateSipDialRules removeSipDialRules	To add, update, get, and remove SipDialRules	dialPattern (Mandatory) name (Mandatory) description patterns plars
updateAnnunciator getAnnunciator	To update, and get Annunciator	name (Mandatory) description devicePoolName locationName useTrustedRelayPoint
addMtp getMtp updateMtp removeMtp	To add, update, get, and remove Mtp	mtpType (Mandatory) name (Mandatory) description devicePoolName (Mandatory) useTrustedRelayPoint
getMohAudioSource updateMohAudioSource removeMohAudioSource	To update, get, and remove MohAudioSource	sourceId (Mandatory) newName sourceFile multicast repeat

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
updateFixedMohAudioSource	To update FixedMohAudioSource	name (Mandatory) multicast enable
getFixedMohAudioSource	To get FixedMohAudioSource	sourceId name multicast enable
addMessageWaiting getMessageWaiting updateMessageWaiting removeMessageWaiting	To add, update, get, and remove MessageWaiting	pattern (Mandatory) routePartitionName (Mandatory) description messageWaitingIndicator (Mandatory) callingSearchSpaceName
addDefaultDeviceProfile getDefaultDeviceProfile updateDefaultDeviceProfile removeDefaultDeviceProfile	To add, update, get, and remove DefaultDeviceProfile	name (Mandatory) description product (Mandatory) class (Mandatory) protocol (Mandatory) protocolSide (Mandatory) userHoldMOHAudioSourceId userLocale phoneButtonTemplate (Mandatory) softkeyTemplate (Mandatory) privacy (Mandatory) singleButtonBarge joinAcrossLines ignorePi dndStatus dndRingSetting dndOption mlppDomainId mlppIndication (Mandatory) preemption (Mandatory) alwaysUsePrimeLine alwaysUsePrimeLineForVoiceMessage emccCallingSearchSpace

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addIpPhoneServices getIpPhoneServices updateIpPhoneServices removeIpPhoneServices	To add, update, get, and remove IpPhoneServices	serviceName (Mandatory) asciiServiceName (Mandatory) serviceDescription serviceUrl (Mandatory) secureServiceUrl serviceCategory (Mandatory) serviceType (Mandatory) serviceVendor serviceVersion enabled enterpriseSubscription parameters parameter (Mandatory) name (Mandatory) displayName (Mandatory) default description (Mandatory) paramRequired paramPassword
addCiscoCatalyst600024PortFXSGateway getCiscoCatalyst600024PortFXSGateway updateCiscoCatalyst600024PortFXSGateway removeCiscoCatalyst600024PortFXSGateway	To add, update, get, and remove CiscoCatalyst600024PortFXSGateway	name(mandatory) description product(mandatory) class(mandatory) protocol(mandatory) protocolSide(mandatory) callingSearchSpaceName devicePoolName(mandatory) commonDeviceConfigName networkLocale locationName(mandatory) mediaResourceListName automatedAlternateRoutingCssName aarNeighborhoodName loadInformation vendorConfig

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		traceFlag mlppDomainId useTrustedRelayPoint cgpnTransformationCssName useDevicePoolCgpnTransformCss geoLocationName ports port(mandatory) portNumber(mandatory) attendantDn unattendedPort callerIdDn callerIdEnable(mandatory) callingPartySelection digitSending expectedDigits(mandatory) sigDigits(mandatory) lines prefixDn presentationBit silenceSuppressionThreshold smdiPortNumber(mandatory) startDialProtocol trunk trunkDirection(mandatory) trunkLevel trunkPadRx trunkPadTx vendorConfig timer1(mandatory) timer2(mandatory) timer3(mandatory) timer4(mandatory) timer5(mandatory) timer6(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		portSelectionOrder(mandatory) transmitUtf8 geoLocationFilterName
addCiscoCatalyst6000T1VoIPGatewayT1 getCiscoCatalyst6000T1VoIPGatewayT1 updateCiscoCatalyst6000T1VoIPGatewayT1 removeCiscoCatalyst6000T1VoIPGatewayT1	To add, update, get, and remove CiscoCatalyst6000T1VoIPGatewayT1	name(mandatory) description product(mandatory) class(mandatory) protocol(mandatory) protocolSide(mandatory) callingSearchSpaceName(mandatory) devicePoolName(mandatory) commonDeviceConfigName(mandatory) networkLocation locationName(mandatory) mediaResourceListName(mandatory) automatedAlternateRoutingCSSName(mandatory) aarNeighborhoodName(mandatory) loadInformation vendorConfig traceFlag mlppDomainId mlppIndicationStatus preemption useTrustedRelayPoint retryVideoCallAsAudio cgpnTransformationCSSName(mandatory) useDevicePoolCgpnTransformCSS geoLocationName(mandatory) sendGeoLocation(mandatory) ports port(mandatory) portNumber(mandatory) attendantDn unattendedPort

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		callerIdDn callerIdEnable(mandatory) callingPartySelection(mandatory) digitSending(mandatory) expectedDigits(mandatory) sigDigits(mandatory) prefixDn(mandatory) presentationBit(mandatory) silenceSuppressionThreshold(mandatory) startDialProtocol trunk(mandatory) trunkDirection(mandatory) trunkLevel(mandatory) trunkPadRx(mandatory) trunkPadTx(mandatory) vendorConfig callerId(mandatory) endpointId timer1 timer2 timer3 timer4 timer5 timer6 trunkSelectionOrder(mandatory) clockReference(mandatory) csuParam(mandatory) digitSending(mandatory) pcmType(mandatory) fdlChannel(mandatory) yellowAlarm(mandatory) zeroSuppression(mandatory) smdiBasePort(mandatory) handlerDtmfPrecedenceSignals cdpnTransformationCssName(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		useDevicePoolCdpnTransformCss geoLocationFilterName(mandatory) pstnAccess imeE164TransformationName
addSrst updateSrst removeSrst	To add, update, and remove Srst	name(mandatory) port(mandatory) ipAddress(mandatory) SipNetwork SipPort(mandatory) isSecure
getSrst	To get Srst	name port ipAddress SipNetwork SipPort srstCertificatePort isSecure
addApplicationUserCapfProfile updateApplicationUserCapfProfile removeApplicationUserCapfProfile	To add, update, and remove ApplicationUserCapfProfile	applicationUser(mandatory) instanceId(mandatory) certificateOperation(mandatory) authenticationMode(mandatory) authenticationString keySize(mandatory) operationCompletion
getApplicationUserCapfProfile	To get ApplicationUserCapfProfile	applicationUser instanceId certificateOperation authenticationMode authenticationString keySize operationCompletion certificateOperationStatus
addEndUserCapfProfile updateEndUserCapfProfile removeEndUserCapfProfile	To add, update, and remove EndUserCapfProfile	endUserId(mandatory) instanceId(mandatory) certificateOperation(mandatory) authenticationMode(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		authenticationString keySize(mandatory) operationCompletion
getEndUserCapfProfile	To get EndUserCapfProfile	endUserId instanceId certificateOperation authenticationMode authenticationString keySize operationCompletion certificateOperationStatus
addUserPhoneAssociation	To add UserPhoneAssociation	userId(mandatory) password pin lastName(mandatory) middleName firstName productType(mandatory) name(mandatory) dnCssName phoneCssName e164Mask(mandatory) extension(mandatory) routePartitionName voiceMailProfileName enableExtensionMobility
addGateway getGateway removeGateway updateGateway	To add, update, get, and remove Gateway	domainName(mandatory) description product(mandatory) protocol(mandatory) callManagerGroupName(mandatory) units unit index(mandatory) product(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		subunits subunit index(mandatory) product(mandatory) beginPort vendorConfig
addGatewayEndpointAnalogAccess getGatewayEndpointAnalogAccess updateGatewayEndpointAnalogAccess removeGatewayEndpointAnalogAccess	To add, update, get, and remove GatewayEndpointAnalogAccess	domainName(mandatory) gatewayUuid(mandatory) unit(mandatory) subunit(mandatory) endpoint(mandatory) index(mandatory) name(mandatory) description product(mandatory) model class(mandatory) protocol(mandatory) protocolSide(mandatory) callingSearchSpaceName devicePoolName(mandatory) commonDeviceConfigName networkLocale locationName(mandatory) mediaResourceListName automatedAlternateRoutingCssName aarNeighborhoodName vendorConfig mlppDomainId useTrustedRelayPoint retryVideoCallAsAudio useDevicePoolCgpnTransformCss geoLocationName geoLocationFilterName port(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		portNumber(mandatory) attendantDn unattendedPort callerIdDn callerIdEnable(mandatory) callingPartySelection digitSending expectedDigits(mandatory) sigDigits(mandatory) lines prefixDn presentationBit silenceSuppressionThreshold smdiPortNumber(mandatory) startDialProtocol trunk trunkDirection(mandatory) trunkLevel trunkPadRx trunkPadTx vendorConfig timer1(mandatory) timer2(mandatory) timer3(mandatory) timer4(mandatory) timer5(mandatory) timer6(mandatory) trunkSelectionOrder(mandatory) transmitUtf8 cdpnTransformationCssName useDevicePoolCdpnTransformCss callingPartyNumberPrefix callingPartyStripDigits callingPartyTransformationCssName hotlineDevice

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		packetCaptureMode packetCaptureDuration pstnAccess imeE164TransformationName imeE164DirectoryNumber
addGatewayEndpointDigitalAccessPri getGatewayEndpointDigitalAccessPri updateGatewayEndpointDigitalAccessPri removeGatewayEndpointDigitalAccessPri	To add, update, get, and remove GatewayEndpointDigitalAccessPri	domainName (Mandatory) gatewayUuid (Mandatory) unit (Mandatory) subunit (Mandatory) endpoint (Mandatory) index (Mandatory) name (Mandatory) description product(Mandatory) class(Mandatory) protocol(Mandatory) protocolSide(Mandatory) callingSearchSpaceName(Mandatory) devicePoolName(Mandatory) commonDeviceConfigName(Mandatory) networkLocation locationName(Mandatory) networkLocale mediaResourceListName(Mandatory) automatedAlternateRoutingCssName(Mandatory) aarNeighborhoodName(Mandatory) loadInformation vendorConfig mlppDomainId mlppIndicationStatus mlppPreemption useTrustedRelayPoint cgpnTransformationCssName(Mandatory) useDevicePoolCgpnTransformCss

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		geoLocationName(Mandatory) redirectInboundNumberIe(Mandatory) calledPlan(Mandatory) calledPri(Mandatory) callerIdDn(Mandatory) callingPartySelection(Mandatory) callingPlan(Mandatory) callingPri(Mandatory) chanIE(Mandatory) clockReference(Mandatory) dChannelEnable(Mandatory) channelSelectionOrder(Mandatory) displayIe(Mandatory) pcmType(Mandatory) csuParam(Mandatory) firstDelay(Mandatory) interfaceIdPresent(Mandatory) interfaceId(Mandatory) intraDelay(Mandatory) mcdnEnable(Mandatory) redirectOutboundNumberIe(Mandatory) numDigitsToStrip(Mandatory) passingPrecedenceLevelThrough(Mandatory) prefix(Mandatory) callingLinePresentationBit(Mandatory) connectedLineIdPresentation(Mandatory) priProtocol(Mandatory) securityAccessLevel(Mandatory) sendCallingNameInFacilityIe(Mandatory) sendExLeadingCharInDispIe(Mandatory) sendRestart(Mandatory) setupNonIsdnPi(Mandatory) sigDigits(Mandatory) span(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		statusPoll(Mandatory) smdiBasePort(Mandatory) GClearEnable(Mandatory) packetCaptureMode packetCaptureDuration transmitUtf8 v150 asn1RoseOidEncoding qsigVariant unattendedPort cdpnTransformationCssName useDevicePoolCdpnTransformCss nationalPrefix internationalPrefix unknownPrefix subscriberPrefix geoLocationFilterName routeClassSignalling nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits nationalTransformationCssName internationalTransformationCssName unknownTransformationCssName subscriberTransformationCssName pstnAccess imeE164TransformationName
addGatewayEndpointDigitalAccessBri getGatewayEndpointDigitalAccessBri updateGatewayEndpointDigitalAccessBri remoevGatewayEndpointDigitalAccessBri	To add, update, get, and remove GatewayEndpointDigitalAccessBri	domainName(Mandatory) gatewayUuid(Mandatory) unit(Mandatory) subunit(Mandatory) endpoint(Mandatory) index(Mandatory) name(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		description product(Mandatory) class(Mandatory) protocol(Mandatory) protocolSide(Mandatory) callingSearchSpaceName(Mandatory) devicePoolName(Mandatory) commonDeviceConfigName(Mandatory) networkLocation locationName(Mandatory) mediaResourceListName(Mandatory) networkLocale automatedAlternateRoutingCssName(Mandatory) aarNeighborhoodName(Mandatory) vendorConfig cgpnTransformationCssName(Mandatory) useDevicePoolCgpnTransformCss geoLocationName(Mandatory) redirectInboundNumberIe(Mandatory) briProtocol(Mandatory) calledPlan(Mandatory) calledPri(Mandatory) callerIdDn(Mandatory) callingPartySelection(Mandatory) callingPlan(Mandatory) callingPri(Mandatory) clockReference(Mandatory) csuParam(Mandatory) dChannelEnable(Mandatory) channelSelectionOrder(Mandatory) pcmType(Mandatory) firstDelay(Mandatory) intraDelay(Mandatory) redirectOutboundNumberIe(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		numDigitsToStrip(Mandatory) prefix(Mandatory) presentationBit(Mandatory) sendRestart(Mandatory) setupNonIsdnPi(Mandatory) sigDigits(Mandatory) statusPoll(Mandatory) packetCaptureMode packetCaptureDuration cdpnTransformationCssName useDevicePoolCdpnTransformCss geoLocationFilterName nationalPrefix internationalPrefix unknownPrefix subscriberPrefix nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits nationalTransformationCssName internationalTransformationCssName unknownTransformationCssName subscriberTransformationCssName pstnAccess imeE164TransformationName
addGatewayEndpointDigitalAccessT1 getGatewayEndpointDigitalAccessT1 removeGatewayEndpointDigitalAccessT1 updateGatewayEndpointDigitalAccessT1	To add, update, get, and remove GatewayEndpointDigitalAccessT1	domainName(mandatory) gatewayUuid(mandatory) unit(mandatory) subUnit(mandatory) endpoint(mandatory) index(mandatory) name(mandatory) description product(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		class(mandatory) protocol(mandatory) protocolSide(mandatory) callingSearchSpaceName(mandatory) devicePoolName(mandatory) commonDeviceConfigName(mandatory) networkLocation locationName(mandatory) mediaResourceListName(mandatory) automatedAlternateRoutingCssName(mandatory) aarNeighborhoodName(mandatory) loadInformation vendorConfig traceFlag mlppDomainId mlppIndicationStatus preemption useTrustedRelayPoint retryVideoCallAsAudio cgpnTransformationCssName(mandatory) useDevicePoolCgpnTransformCss geoLocationName(mandatory) sendGeoLocation(mandatory) cdpnTransformationCssName(mandatory) useDevicePoolCdpnTransformCss v150 geoLocationFilterName ports port portNumber(mandatory) attendantDn unattendedPort callerIdDn callerIdEnable(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		callingPartySelection(mandatory) digitSending(mandatory) expectedDigits(mandatory) sigDigits(mandatory) prefixDn(mandatory) presentationBit(mandatory) silenceSuppressionThreshold(mandatory) startDialProtocol(mandatory) trunk(mandatory) trunkDirection(mandatory) trunkLevel(mandatory) trunkPadRx(mandatory) trunkPadTx(mandatory) vendorConfig callerId endPointId timer1 timer2 timer3 timer4 timer5 timer6 trunkSelectionOrder(mandatory) clockReference(mandatory) csuParam(mandatory) digitSending(mandatory) pcmType(mandatory) fdlChannel(mandatory) yellowAlarm(mandatory) zeroSuppresion(mandatory) smdiBasePort(mandatory) handleDtmfPrecedenceSignals encodeOutboundVoiceRouteClass routeClassSignalling(mandatory) pstnAccess imeE164TransformationName

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addCiscoCatalyst6000E1VoIPGateway getCiscoCatalyst6000E1VoIPGateway updateCiscoCatalyst6000E1VoIPGateway removeCiscoCatalyst6000E1VoIPGateway	To add, update, get, and remove CiscoCatalyst6000E1VoIPGateway	name(mandatory) description product(mandatory) class(mandatory) protocol(mandatory) protocolSide(mandatory) callingSearchSpaceName(mandatory) devicePoolName(mandatory) commonDeviceConfigName(mandatory) networkLocation locationName(mandatory) networkLocale mediaResourceListName(mandatory) automatedAlternateRoutingCssName(mandatory) aarNeighborhoodName(mandatory) loadInformation vendorConfig mlppDomainId useTrustedRelayPoint cgpnTransformationCssName(mandatory) useDevicePoolCgpnTransformCss geoLocationName(mandatory) redirectInboundNumberIe(mandatory) calledPlan(mandatory) calledPri(mandatory) callerIdDn(mandatory) callingPartySelection(mandatory) callingPlan(mandatory) callingPri(mandatory) chanIe(mandatory) clockReference(mandatory) dChannelEnable(mandatory) channelSelectionOrder(mandatory) displayIE(mandatory) pcmType(mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		csuParam(mandatory) firstDelay(mandatory) interfaceIdPresent(mandatory) interfaceId(mandatory) intraDelay(mandatory) mcdnEnable(mandatory) redirectOutboundNumberIe(mandatory) numDigitsToStrip(mandatory) passingPrecedenceLevelThrough(mandatory) prefix(mandatory) callingLinePresentationBit(mandatory) connectedLineIdPresentation(mandatory) priProtocol(mandatory) securityAccessLevel(mandatory) sendCallingNameInFacilityIe(mandatory) sendExLeadingCharInDispIe(mandatory) sendRestart(mandatory) setupNonIsdnPi(mandatory) sigDigits(mandatory) span(mandatory) statusPoll(mandatory) smdiBasePort(mandatory) packetCaptureMode packetCaptureDuration transmitUtf8 v150 asn1RoseOidEncoding QSIGVariant unattendedPort cdpnTransformationCssName useDevicePoolCdpnTransformCss nationalPrefix internationalPrefix unknownPrefix

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		subscriberPrefix geoLocationFilterName nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits nationalTransformationCssName internationalTransformationCssName unknownTransformationCssName subscriberTransformationCssName useDevicePoolCgpnTransformCssNatl useDevicePoolCgpnTransformCssIntl useDevicePoolCgpnTransformCssUnkn useDevicePoolCgpnTransformCssSubs pstnAccess imeE164TransformationName
addCiscoCatalyst6000T1VoIPGatewayPri getCiscoCatalyst6000T1VoIPGatewayPri updateCiscoCatalyst6000T1VoIPGatewayPri removeCiscoCatalyst6000T1VoIPGatewayPri	To add, update, get, and remove CiscoCatalyst6000T1VoIPGatewayPri	name(Mandatory) description product(Mandatory) class(Mandatory) protocol(Mandatory) protocolSide(Mandatory) callingSearchSpaceName(Mandatory) devicePoolName(Mandatory) commonDeviceConfigName(Mandatory) networkLocation locationName(Mandatory) networkLocale mediaResourceListName(Mandatory) automatedAlternateRoutingCssName(Mandatory) aarNeighborhoodName(Mandatory) loadInformation vendorConfig mlppDomainId mlppIndicationStatus

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		mlppPreemption useTrustedRelayPoint cgpnTransformationCssName(Mandatory) useDevicePoolCgpnTransformCss geoLocationName(Mandatory) redirectInboundNumberIe(Mandatory) calledPlan(Mandatory) calledPri(Mandatory) callerIdDn(Mandatory) callingPartySelection(Mandatory) callingPlan(Mandatory) callingPri(Mandatory) chanIe(Mandatory) clockReference(Mandatory) dChannelEnable(Mandatory) channelSelectionOrder(Mandatory) displayIE(Mandatory) pcmType (Mandatory) csuParam(Mandatory) firstDelay(Mandatory) interfaceIdPresent(Mandatory) interfaceId(Mandatory) intraDelay(Mandatory) mcdnEnable(Mandatory) redirectOutboundNumberIe(Mandatory) numDigitsToStrip(Mandatory) passingPrecedenceLevelThrough(Mandatory) prefix(Mandatory) callingLinePresentationBit(Mandatory) connectedLineIdPresentation(Mandatory) priProtocol(Mandatory) securityAccessLevel(Mandatory) sendCallingNameInFacilityIe(Mandatory) sendExLeadingCharInDispIe(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		sendRestart(Mandatory) setupNonIsdnPi(Mandatory) sigDigits(Mandatory) span(Mandatory) statusPoll(Mandatory) smdiBasePort(Mandatory) packetCaptureMode packetCaptureDuration transmitUtf8 v150 asn1RoseOidEncoding QSIGVariant unattendedPort cdpnTransformationCssName useDevicePoolCdpnTransformCss nationalPrefix internationalPrefix unknownPrefix subscriberPrefix geoLocationFilterName nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits nationalTransformationCssName internationalTransformationCssName unknownTransformationCssName subscriberTransformationCssName useDevicePoolCgpnTransformCssNatl useDevicePoolCgpnTransformCssIntl useDevicePoolCgpnTransformCssUnkn useDevicePoolCgpnTransformCssSubs pstnAccess imeE164TransformationName

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addCiscoCatalyst6000T1VoIPGatewayT1 getCiscoCatalyst6000T1VoIPGatewayT1 updateCiscoCatalyst6000T1VoIPGatewayT1 removeCiscoCatalyst6000T1VoIPGatewayT1	To add, update, get, and remove CiscoCatalyst6000T1VoIPGatewayT1	name(Mandatory) description product(Mandatory) class(Mandatory) protocol(Mandatory) protocolSide(Mandatory) callingSearchSpaceName(Mandatory) devicePoolName(Mandatory) commonDeviceConfigName(Mandatory) networkLocation locationName(Mandatory) mediaResourceListName(Mandatory) automatedAlternateRoutingCssName(Mandatory) aarNeighborhoodName(Mandatory) loadInformation vendorConfig traceFlag mlppDomainIdmlppIndicationStatus preemption useTrustedRelayPoint retryVideoCallAsAudio cgpnTransformationCssName(Mandatory) useDevicePoolCgpnTransformCss geoLocationName(Mandatory) sendGeoLocation(Mandatory) ports port(Mandatory) portNumber(Mandatory) attendantDn unattendedPort callerIdDn callerIdEnable(Mandatory) callingPartySelection(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		digitSending(Mandatory) expectedDigits(Mandatory) sigDigits(Mandatory) prefixDn(Mandatory) presentationBit(Mandatory) silenceSuppressionThreshold(Mandatory) startDialProtocol(Mandatory) trunk(Mandatory) trunkDirection(Mandatory) trunkLevel(Mandatory) trunkPadRx(Mandatory) trunkPadTx(Mandatory) vendorConfig callerId(Mandatory) endpointId timer1 timer2 timer3 timer4 timer5 timer6 trunkSelectionOrder(Mandatory) clockReference(Mandatory) csuParam(Mandatory) digitSending(Mandatory) pcmType(Mandatory) fdlChannel(Mandatory) yellowAlarm(Mandatory) zeroSupression(Mandatory) smdiBasePort(Mandatory) handleDtmfPrecedenceSignals cdpnTransformationCssName(Mandatory) useDevicePoolCdpnTransformCss geoLocationFilterName(Mandatory)

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		pstnAccess imeE164TransformationName
listRoutePlan	To list RoutePlan	dnOrPattern partition type routeDetail
addCallingPartyTransformationPattern updateCallingPartyTransformationPattern removeCallingPartyTransformationPattern	To add, update, and remove CallingPartyTransformationPattern	pattern(mandatory) description routePartitionName(mandatory) callingPartyTransformationMask useCallingPartyPhoneMask dialPlanName digitDiscardInstructionName callingPartyPrefixDigits routeFilterName callingLinePresentationBit callingPartyNumberingPlan callingPartyNumberType
getCallingPartyTransformationPattern	To get CallingPartyTransformationPattern	pattern description usage routePartitionName callingPartyTransformationMask useCallingPartyPhoneMask dialPlanName digitDiscardInstructionName patternUrgency callingPartyPrefixDigits routeFilterName callingLinePresentationBit callingPartyNumberingPlan callingPartyNumberType

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addUnitsToGateway removeUnitsToGateway	To add, and removeUnitsToGateway	domainName(mandatory) gatewayUuid(mandatory) units(mandatory) unit(mandatory) index(mandatory) product(mandatory) subunits subunit(mandatory) index(mandatory) product(mandatory) beginPort
addGatewaySubunits removeGatewaySubunits	To add, and remove GatewaySubunits	domainName(mandatory) gatewayUuid(mandatory) unit(mandatory) subunits(mandatory) subunit(mandatory) index(mandatory) product(mandatory) beginPort
addLdapDirectory getLdapDirectory updateLdapDirectory removeLdapDirectory	To add, update, get, and remove LdapDirectory	name(mandatory) ldapDn(mandatory) ldapPassword(mandatory) userSearchBase(mandatory) repeatable intervalValue(mandatory) scheduleUnit nextExecTime(mandatory) servers(mandatory) server(mandatory) hostName(mandatory) ldapPortNumber(mandatory) sslEnabled middleName phoneNumber mailId

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
		ldapFilter
addSafCcdPurgeBlockLearnedRoutes getSafCcdPurgeBlockLearnedRoutes updateSafCcdPurgeBlockLearnedRoutes removeSafCcdPurgeBlockLearnedRoutes	To add, update, get, and remove SafCcdPurgeBlockLearnedRoutes	learnedPattern learnedPatternPrefix callControlIdentity ipAddress
addVpnGateway getVpnGateway updateVpnGateway removeVpnGateway	To add, update, get, and remove VpnGateway	name(mandatory) description url(mandatory) certificates(mandatory) certificate(mandatory) issuerName(mandatory) serialNumber(mandatory)
addVpnGroup getVpnGroup updateVpnGroup removeVpnGroup	To add, update, get, and remove VpnGroup	name(mandatory) description vpnGateways vpnGateway vpnGatewayName(mandatory) priority(mandatory)
addVpnProfile getVpnProfile updateVpnProfile removeVpnProfile	To add, update, get, and remove VpnProfile	name(mandatory) description autoNetworkDetection mtu failToConnect clientAuthentication pwdPersistant enableHostIdCheck

Table 2-3 New Operations in Unified CM 8.0(1) (continued)

Operation	Purpose	Added Tags
addTransformationProfile getTransformationProfile updateTransformationProfile removeTransformationProfile	To add, update, get, and remove TransformationProfile	name(mandatory) description nationalStripDigits internationalStripDigits unknowmStripDigits subscriberStripDigits nationalPrefix internationalPrefix unknownPrefix subscriberPrefix nationalCssName internationalCssName unknownCssName subscriberCssName
addLdapFilter getLdapFilter updateLdapFilter removeLdapFilter	To add, update, get, and remove LdapFilter	name(mandatory) filter(mandatory)
addAppServerInfo getAppserverInfo updateAppserverInfo removeAppServerInfo	To add, update, get, and remove AppServerInfo	appServerName(mandatory) appServerContent(mandatory) content

Changed APIs

Table 2-4 describes changed operations in Unified CM 8.0(1).

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
AARGroup		AarGroup	
	—		aarGroupFromName
	—		aarGroupToName
	name		name
	—		prefixDigit
	—		relatedGroup
	—		relatedGroups

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
AARGroupMatrix		AarGroupMatrix	
	—		aarGroupFromName
	—		aarGroupToName
	prefixDigit		prefixDigit
AppUser		AppUser	
	acceptOutOfDialogRefer		acceptOutOfDialogRefer
	acceptPresenceSubscription		acceptPresenceSubscription
	acceptUnsolicitedNotification		acceptUnsolicitedNotification
	allowReplaceHeader		allowReplaceHeader
	—		associatedCapfProfiles
	—		associatedCapfProfiles
	associatedDevices		associatedDevices
	associatedUserGroups		associatedGroups
	—		capfProfileInstanceId
	—		ctiControlledDeviceProfiles
	device		device
	—		deviceProfile
	digestCredentials		digestCredentials
	—		isStandard
	name		name
	password		password
	—		passwordCredentials
	presenceGroupName		presenceGroupName
	—		pwdCredDoesNotExpire
	—		pwdCredLockedByAdministrator
	—		pwdCredPolicyName
	—		pwdCredTimeAdminLockout
	—		pwdCredTimeChanged
	—		pwdCredUserCantChange
	—		pwdCredUserMustChange
	userGroup		userGroup
	userid		userid
	—		userRole
	—		userRoles

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
CalledPartyTransformationPattern		CalledPartyTransformationPattern	
	calledPartyNumberingPlan		calledPartyNumberingPlan
	calledPartyNumberType		calledPartyNumberType
	calledPartyPrefixDigits		calledPartyPrefixDigits
	calledPartyTransformationMask		calledPartyTransformationMask
	description		description
	dialPlanName		dialPlanName
	digitDiscardInstructionName		digitDiscardInstructionName
	pattern		pattern
	patternUrgency		patternUrgency
	routeFilterName		routeFilterName
	routePartitionName		routePartitionName
	usage		usage
CallerFilterList		CallerFilterList	
	callerFilterMask		callerFilterMask
	description		description
	dnMask		DnMask
	endUserIdName		endUserIdName
	isAllowedType		isAllowedType
	member		member
	members		members
	name		name
CallManager		CallManager	
	—		autoRegistration
	—		e164Mask
	—		endDn
	—		ethernetPhonePort
	description		description
	—		keepAlive
	—		listen
	—		mgcpPorts
	name		name
	—		nextDn
	—		ports
	—		processNodeName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		routePartitionName
	—		sipPhonePort
	—		sipPhoneSecurePort
	—		sipPorts
	—		startDn
	—		autoRegistration
CallManagerGroup		CallManagerGroup	
	callManagerName		callManagerName
	member		member
	members		members
	name		name
	—		priority
	—		tftpDefault
CallPark		CallPark	
	callManagerName		callManagerName
	description		description
	pattern		pattern
	routePartitionName		routePartitionName
	usage		usage
CallPickupGroup		CallPickupGroup	
	calledPartyInfo		calledPartyInfo
	callInfoForPickupNotification		callInfoForPickupNotification
	callingPartyInfo		callingPartyInfo
	description		description
	—		dnPattern
	—		members
	—		member
	name		name
	pattern		pattern
	—		pickupDnAndPartition
	—		pickupGroupName
	pickupNotification		pickupNotification
	pickupNotificationTimer		pickupNotificationTimer
	—		priority
	routePartitionName		routePartitionName
	—		routePartitionName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	usage		usage
CMCInfo		CmcInfo	
	code		code
	description		description
CommonDeviceConfig		CommonDeviceConfig	
	allowAutoConfigurationForPhones		allowAutoConfigurationForPhones
	IPAddressingMode		ipAddressingMode
	IPAddressingModePreferenceControl		ipAddressingModePreferenceControl
	mlppDomainId		mlppDomainId
	mlppIndicationStatus		mlppIndicationStatus
	name		name
	networkHoldMOHAudioSourceId		networkHoldMohAudioSourceId
	preemption		preemption
	softkeyTemplateName		softkeyTemplateName
	—		useImeForOutboundCalls
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userLocale		userLocale
	useTrustedRelayPoint		useTrustedRelayPoint
CommonPhoneConfig		CommonPhoneConfig	
	alwaysUsePrimeLine		alwaysUsePrimeLine
	alwaysUsePrimeLineforVoiceMessage		alwaysUsePrimeLineForVoiceMessage
	backgroundImage		backgroundImage
	ciscoCamera		—
	description		description
	dndAlertingType		dndAlertingType
	dndOption		dndOption
	name		name
	phonePersonalization		phonePersonalization
	phoneServiceDisplay		phoneServiceDisplay
	sdio		—
	sshPwd		sshPwd
	sshUserId		sshUserId
	unlockPwd		unlockPwd

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	usb1		—
	usb2		—
	usbClasses		—
	vendorConfig		vendorConfig
	—		vpnGroupName
	—		vpnProfileName
ConferenceBridge		ConferenceBridge	
	—		commonDeviceConfigName
	description		description
	devicePoolName		devicePoolName
	loadInformation		loadInformation
	locationName		locationName
	maximumCapacity		maximumCapacity
	name		name
	product		product
	—		securityProfileName
	subUnit		subUnit
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		vendorConfig
CredentialPolicy		CredentialPolicy	
	administratorMustUnlock		—
	credChangeDuration		credChangeDuration
	credExpiresAfter		credExpiresAfter
	expiryWarningDays		expiryWarningDays
	failedLogon		failedLogon
	inactiveDaysAllowed		inactiveDaysAllowed
	lockoutDuration		lockoutDuration
	minCredLength		minCredLength
	name		name
	neverExpires		—
	noLimitForFailedLogon		—
	prevCredStoredNum		prevCredStoredNum
	resetFailedLogonAttempts		resetFailedLogonAttempts
	trivialCredCheck		trivialCredCheck
CSS		Css	
	clause		clause

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	description		description
	dialPlanWizardGenId		dialPlanWizardGenId
	—		index
	member		member
	members		members
	name		name
	partitionUsage		partitionUsage
	routePartitionName		routePartitionName
CTIRoutePoint		CtiRoutePoint	
	aarNeighborhoodName		—
	—		asciiLabel
	—		associatedEndusers
	—		audibleMwi
	automatedAlternateRoutingCSSName		—
	—		busyTrigger
	—		callerName
	—		callerNumber
	—		callInfoDisplay
	callingSearchSpaceName		callingSearchSpaceName
	cgpnTransformationCSSName		cgpnTransformationCssName
	class		class
	commonDeviceConfigName		commonDeviceConfigName
	—		consecutiveRingSetting
	description		description
	devicePoolName		devicePoolName
	—		dialedNumber
	—		dialPlanWizardId
	—		directoryNumber
	—		dirn
	—		display
	—		displayAscii
	—		e164Mask
	—		enduser
	—		geoLocationName
	—		index

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		label
	—		line
	—		lineIdentifier
	—		lines
	loadInformation		—
	locationName		locationName
	—		maxNumCalls
	mediaResourceListName		mediaResourceListName
	mlppDomainId		—
	mlppIndicationStatus		—
	model		model
	—		monitoringCssName
	—		mwlPolicy
	name		name
	networkHoldMOHAudioSourceId		networkHoldMohAudioSourceId
	networkLocation		—
	—		partitionUsage
	preemption		—
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		recordingFlag
	—		recordingProfileName
	—		redirectedNumber
	retryVideoCallAsAudio		—
	—		ringSetting
	—		ringSettingActivePickupAlert
	—		ringSettingIdlePickupAlert
	—		routePartitionName
	—		speedDial
	traceFlag		—
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	—		userLocale
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	versionStamp		—
DDI		Ddi	
	clause		clause
	dialPlanName		dialPlanName
	dialPlanTagName		dialPlanTagName
	digitAnalysisId		digitAnalysisId
	member		member
	members		members
	name		name
DeviceMobility		DeviceMobility	
	devicePoolName		devicePoolName
	member		member
	members		members
	name		name
	subNet		subNet
	subNetMaskSz		subNetMaskSz
DeviceMobilityGroup		DeviceMobilityGroup	
	—		description
	name		name
DevicePool		DevicePool	
	aarNeighborhoodName		aarNeighborhoodName
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	autoSearchSpaceName		autoSearchSpaceName
	—		calledPartyUnknownTransformationCssName
	—		callingPartyInternationalPrefix
	—		callingPartyInternationalStripDigits
	—		callingPartyInternationalTransformationCssName
	—		callingPartyNationalPrefix
	—		callingPartyNationalStripDigits
	—		callingPartyNationalTransformationCssName
	—		callingPartySubscriberPrefix
	—		callingPartySubscriberStripDigits

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		callingPartySubscriberTransformationCssName
	—		callingPartyUnknownPrefix
	—		callingPartyUnknownStripDigits
	—		callingPartyUnknownTransformationCssName
	callManagerGroupName		callManagerGroupName
	cdpnTransformCSSName		cdpnTransformationCssName
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformCSSName		cgpnTransformationCssName
	—		cntdPnTransformationCssName
	connectionMonitorDuration		connectionMonitorDuration
	dateTimeSettingName		dateTimeSettingName
	deviceMobilityGroupName		deviceMobilityGroupName
	—		geoLocationFilterName
	—		geoLocationName
	—		imeEnrolledPatternGroupName
	internationalPrefix		—
	internationalStripDigits		—
	joinAcrossLines		joinAcrossLines
	localRouteGroupName		localRouteGroupName
	locationName		locationName
	mediaResourceListName		mediaResourceListName
	mobilityCSSName		mobilityCssName
	name		name
	nationalPrefix		—
	nationalStripDigits		—
	networkLocale		networkLocale
	physicalLocation		physicalLocationName
	regionName		regionName
	revertPriority		revertPriority

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	singleButtonBarge		singleButtonBarge
	srstName		srstName
	subscriberPrefix		—
	subscriberStripDigits		—
	unknownPrefix		—
	unknownStripDigits		—
DeviceProfile		DeviceProfile	
	aarDestinationMask		—
	aarKeepCallHistory		—
	aarNeighborhoodName		—
	aarNeighborhoodName		—
	aarVoiceMailEnabled		—
	addOnModule		addOnModule
	addOnModules		addOnModules
	—		addressMode
	alertingName		—
			allowAutoConfig
	allowCtiControlFlag		—
	alwaysUsePrimeLine		alwaysUsePrimeLine
			alwaysUsePrimeLine
	alwaysUsePrimeLineforVoiceMessage		alwaysUsePrimeLineForVoiceMessage
	—		alwaysUsePrimeLineForVoiceMessage
	—		asciiLabel
	—		asciiLabel
	—		asciiLabel
	—		associatedBlfSdFeatures
	—		associatedEndusers
	associatedPC		—
	—		audibleMwi
	authenticationString		—
	authenticationURL		—
	autoAnswer		—
	blfDest		blfDest
	—		blfDirectedCallPark
	—		blfDirectedCallParks

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	blfDirn		blfDirn
	builtInBridgeStatus		—
	busyLampField		busyLampField
	busyLampFields		busyLampFields
	busyTrigger		busyTrigger
	callerName		callerName
	callerNumber		callerNumber
	callForwardAll		—
	callForwardAlternateParty		—
	callForwardBusy		—
	callForwardBusyInt		—
	callForwardNoAnswer		—
	callForwardNoAnswerInt		—
	callForwardNoCoverage		—
	callForwardNoCoverageInt		—
	callForwardNotRegistered		—
	callForwardNotRegisteredInt		—
	callForwardOnFailure		—
	callInfoDisplay		callInfoDisplay
	callInfoPrivacyStatus		callInfoPrivacyStatus
	—		callInfoPrivacyStatus
	callPickupGroupName		—
	certificateOperation		—
	certificateStatus		—
	cfaCSSClause		—
	cfaCSSPolicy		—
	class		class
	—		class
	commonDeviceConfig		
	consecutiveRingSetting		consecutiveRingSetting
	—		currentConfig
	—		currentProfileName
	—		defaultProfileName
	description		description
	deviceMobilityMode		—
	—		deviceName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	devicePoolName		—
	—		deviceProtocol
	dialedNumber		dialedNumber
	dialPlanWizardId		dialPlanWizardId
	—		directedCallParkDnAndPartition
	—		directedCallParkId
	—		directoryNumber
	directoryURL		—
	dirn		dirn
	dirn		dirn
	display		display
	—		displayAscii
	dndOption		dndOption
	—		dndOption
	dndRingSetting		dndRingSetting
	—		dndRingSetting
	—		dndStatus
	—		dndStatus
	—		dnPattern
	e164Mask		e164Mask
	—		emccCallingSearchSpace
	—		emccCallingSearchSpaceName
	—		enduser
	—		feature
	—		geolocationInfo
	hlogStatus		—
	hrDuration		—
	hrInterval		—
	idleTimeout		—
	idleURL		—
	ignorePresentationIndicators		ignorePresentationIndicators
	—		ignorePresentationIndicators
	—		index
	—		index
	—		index
	—		index

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		index
	informationURL		—
	isActive		—
	—		joinAcrossLines
	—		joinAcrossLines
	label		label
	label		label
	label		label
	—		label
	—		labelAscii
	line		line
	—		lineIdentifier
	lines		lines
	loadInformation		loadInformation
	loadInformation		—
	locationName		—
	—		loginDuration
	—		loginTime
	—		loginUserId
	maxNumCalls		maxNumCalls
	messagesURL		—
	mlppDomainId		mlppDomainId
	—		mlppDomainId
	mlppIndicationStatus		mlppIndicationStatus
	—		mlppIndicationStatus
	model		model
	model		model
	—		model
	—		monitoringCssName
	mwlPolicy		mwlPolicy
	name		name
	—		name
	networkHoldMOHAudioSourceId		—
	networkHoldMOHAudioSourceId		—
	networkLocale		—
	networkLocation		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	numberOfButtons		numberOfButtons
	ownerUserId		—
	packetCaptureDuration		—
	packetCaptureMode		—
	—		partitionUsage
	pattern		—
	patternPrecedence		—
	—		phoneService
	—		phoneServiceCategory
	phoneSuite		—
	phoneTemplateName		phoneTemplateName
	—		phoneTemplateName
	preemption		preemption
	—		preemption
	presenceGroupName		—
	presenceGroupName		—
	—		priority
	product		product
	—		product
	protocol		protocol
	protocolSide		protocolSide
	proxyServerURL		—
	—		recordingFlag
	—		recordingProfileName
	redirectedNumber		redirectedNumber
	releaseCause		—
	remoteDevice		—
	—		remoteSipSrstIp
	—		remoteSipSrstPort
	—		remoteSrstIp
	—		remoteSrstOption
	—		remoteSrstPort
	retryVideoCallAsAudio		—
	ringSetting		ringSetting
	—		ringSettingActivePickupAlert
	—		ringSettingIdlePickupAlert

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		routePartition
	routePartitionName		routePartitionName
	—		routePartitionName
	—		service
	—		serviceNameAscii
	—		services
	servicesURL		—
	shareLineAppearanceCSSName		—
	—		singleButtonBarge
	—		singleButtonBarge
	softkeyTemplateName		softkeyTemplateName
	—		softkeyTemplateName
	speeddial		speeddial
	—		speedDial
	speeddials		speeddials
	—		telecasterServiceName
	traceFlag		traceFlag
	upgradeFinishTime		—
	—		url
	—		urlButtonIndex
	—		urlLabel
	—		urlLabelAscii
	usage		—
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userLocale		userlocale
	—		vendor
	vendorConfig		vendorConfig
	—		version
	versionStamp		versionStamp
	voiceMailProfileName		—
DHCPServer		DhcpServer	
	arpCacheTimeout		arpCacheTimeout
	bootstrapServeripAddress		bootstrapServerIpAddress
	domainName		domainName
	ipAddressLeaseTime		ipAddressLeaseTime

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	primaryDNSipAddress		primaryDnsIpAddress
	primaryTFTPServeripAddress		primaryTftpServerIpAddress
	processNodeName		processNodeName
	rebindingTime		rebindingTime
	renewalTime		renewalTime
	secondaryDNSipAddress		secondaryDnsIpAddress
	secondaryTFTPServeripAddress		secondaryTftpServerIpAddress
	tftpServerName		tftpServerName
DHCPSubnet		DhcpSubnet	
	arpCacheTimeout		arpCacheTimeout
	bootstrapServerIPAddress		bootstrapServerIpAddress
	dhcpServerName		dhcpServerName
	domainName		domainName
	ipAddressLeaseTime		ipAddressLeaseTime
	primaryDNSIpAddress		primaryDnsIpAddress
	primaryEndIPAddress		primaryEndIpAddress
	primaryRouterIPAddress		primaryRouterIpAddress
	primaryStartIPAddress		primaryStartIpAddress
	primaryTFTPServerIPAddress		primaryTftpServerIpAddress
	rebindingTime		rebindingTime
	renewalTime		renewalTime
	secondaryDNSIpAddress		secondaryDnsIpAddress
	secondaryEndIPAddress		secondaryEndIpAddress
	secondaryRouterIPAddress		secondaryRouterIpAddress
	secondaryStartIPAddress		secondaryStartIpAddress
	secondaryTFTPServerIPAddress		secondaryTftpServerIpAddress
	subnetIPAddress		subnetIpAddress
	subnetMask		subnetMask
	tftpServerName		tftpServerName
DialPlan		DialPlan	
	description		description
	name		name
DialPlanTag		DialPlanTag	
	dialPlanName		dialPlanName
	name		name
	operator		operator

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	suppressFromRouteFilter		suppressFromRouteFilter
DirectedCallPark		DirectedCallPark	
	description		description
	pattern		pattern
	retrievalPrefix		retrievalPrefix
	reversionPattern		reversionPattern
	revertCSSName		revertCssName
	routePartitionName		routePartitionName
	usage		usage
FACInfo		FacInfo	
	authorizationLevel		authorizationLevel
	code		code
	name		name
Gatekeeper		Gatekeeper	
	description		description
	enableDevice		enableDevice
	name		name
	retryTimeout		retryTimeout
	rrqTimeToLive		rrqTimeToLive
GeoLocation		GeoLocation	
	cityDivision		cityDivision
	communityName		communityName
	country		country
	description		description
	district		district
	floor		floor
	houseNumber		houseNumber
	houseNumberSuffix		houseNumberSuffix
	landmark		landmark
	leadingStreetDirection		leadingStreetDirection
	location		location
	name		name
	nationalSubDivision		nationalSubDivision
	neighbourhood		neighbourhood
	occupantName		occupantName
	postalCode		postalCode

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	street		street
	streetSuffix		streetSuffix
	trailingStreetSuffix		trailingStreetSuffix
GeoLocationFilter		GeoLocationFilter	
	description		description
	name		name
	useCityDivision		useCityDivision
	useCommunityName		useCommunityName
	useCountry		useCountry
	useDistrict		useDistrict
	—		useFloor
	useHouseNumber		useHouseNumber
	useHouseNumberSuffix		useHouseNumberSuffix
	useLandmark		useLandmark
	useLeadingStreetDirection		useLeadingStreetDirection
	—		useLocation
	useNationalSubDivision		useNationalSubDivision
	useNeighbourhood		useNeighbourhood
	—		useOccupantName
	—		usePostalCode
	useStreet		useStreet
	useStreetSuffix		useStreetSuffix
	useTrailingStreetSuffix		useTrailingStreetSuffix
GeoLocationPolicy		GeoLocationPolicy	
	cityDivision		cityDivision
	communityName		communityName
	country		country
	description		description
	district		district
	floor		floor
	—		geoLocationDeviceA
	—		geoLocationDeviceB
	—		geoLocationPolicyAName
	—		geoLocationPolicyBName
	houseNumber		houseNumber
	houseNumberSuffix		houseNumberSuffix

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	landmark		landmark
	leadingStreetDirection		leadingStreetDirection
	location		location
	—		logicalPartitionPolicy
	name		name
	nationalSubDivision		nationalSubDivision
	neighbourhood		neighbourhood
	occupantName		occupantName
	postalCode		postalCode
	—		relatedPolicies
	—		relatedPolicy
	street		street
	streetSuffix		streetSuffix
	trailingStreetSuffix		trailingStreetSuffix
H323Gateway		H323Gateway	
	aarNeighborhoodName		aarNeighborhoodName
	AllowH235PassThrough		allowH235PassThrough
	ASN1ROSEOIDEncoding		asn1RoseOidEncoding
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	calledNumberingPlan		calledNumberingPlan
	calledPartyIENumberType		calledPartyIeNumberType
	—		calledPartyInternationalPrefix
	—		calledPartyInternationalStripDigits
	—		calledPartyInternationalTransformationCssName
	—		calledPartyNationalPrefix
	—		calledPartyNationalStripDigits
	—		calledPartyNationalTransformationCssName
	—		calledPartySubscriberPrefix
	—		calledPartySubscriberStripDigits
	—		calledPartySubscriberTransformationCssName
	—		calledPartyUnknownStripDigits
	—		calledPartyUnknownTransformationCssName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	callerIdDN		callerIdDn
	callingLineIdPresentation		callingLineIdPresentation
	callingNumberingPlan		callingNumberingPlan
	callingPartyIENumberType		callingPartyIeNumberType
	—		callingPartyInternationalPrefix
	—		callingPartyInternationalStripDigits
	—		callingPartyInternationalTransformationCssName
	—		callingPartyNationalPrefix
	—		callingPartyNationalStripDigits
	—		callingPartyNationalTransformationCssName
	callingPartySelection		callingPartySelection
	—		callingPartySubscribercalledPartyUnknownPrefix
	—		callingPartySubscriberPrefix
	—		callingPartySubscriberStripDigits
	—		callingPartyUnknownPrefix
	—		callingPartyUnknownStripDigits
	—		callingPartyUnknownTransformationCssName
	callingSearchSpaceName		callingSearchSpaceName
	cdpnTransformationCSSName		cdpnTransformationCssName
	cgpnTransformationCSSName		cgpnTransformationCssName
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	class		class
	—		codecForOutboundFaststart
	commonDeviceConfigName		commonDeviceConfigName
	description		description
	devicePoolName		devicePoolName
	displayIEDelivery		displayIeDelivery

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	enableInboundFaststart		enableInboundFaststart
	enableOutboundFaststart		enableOutboundFaststart
	—		geoLocationFilterName
	—		geoLocationName
	—		imeE164TransformationName
	—		inalSet
	internationalPrefix		—
	internationalStripDigits		—
	licensedCapacity		—
	loadInformation		loadInformation
	locationName		locationName
	mediaResourceListName		mediaResourceListName
	mlppDomainId		mlppDomainId
	mlppIndicationStatus		—
	model		model
	mtpRequired		mtpRequired
	name		name
	nationalPrefix		—
	nationalStripDigits		—
	networkLocation		networkLocation
	packetCaptureDuration		packetCaptureDuration
	packetCaptureMode		packetCaptureMode
	—		pathReplacementSupport
	preemption		—
	prefixDN		prefixDn
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		pstnAccess
	QSIGVariant		qsigVariant
	redirectInboundNumberIE		redirectInboundNumberIe
	redirectOutboundNumberIE		redirectOutboundNumberIe
	retryVideoCallAsAudio		retryVideoCallAsAudio
	significantDigits		sigDigits
	—		signalingPort
	srtpAllowed		srtpAllowed

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	subscriberPrefix		—
	subscriberStripDigits		—
	traceFlag		traceFlag
	—		TransformationCssName
	—		transmitUtf8
	tunneledProtocol		tunneledProtocol
	unknownPrefix		—
	unknownStripDigits		—
	useDevicePoolCdpnTransformCSS		useDevicePoolCdpnTransformCss
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		vendorConfig
	versionStamp		—
	waitForFarEndH245TerminalSet		waitForFarEndH245Term
H323Phone		H323Phone	
	aarNeighborhoodName		aarNeighborhoodName
	—		alwaysUsePrimeLine
	—		alwaysUsePrimeLineForVoiceMessage
	—		asciiLabel
	—		associatedEndusers
	—		audibleMwi
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	—		busyTrigger
	callerIdDN		callerIdDn
	—		callerName
	—		callerNumber
	—		callInfoDisplay
	callingLineIdPresentation		callingLineIdPresentation
	callingPartySelection		callingPartySelection
	callingSearchSpaceName		callingSearchSpaceName
	—		cgpnTransformationCssName
	class		class
	codecForOutboundFaststart		—
	commonDeviceConfigName		commonDeviceConfigName
	—		commonPhoneConfigName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	description		description
	devicePoolName		devicePoolName
	—		dialedNumber
	—		dialPlanWizardId
	—		directoryNumber
	—		dirn
	—		display
	—		displayAscii
	displayIEDelivery		displayIEDelivery
	e164		e164
	—		e164Mask
	enableInboundFaststart		—
	enableOutboundFaststart		—
	—		enduser
	gatekeeperInfo		gateKeeperInfo
	gatekeeperName		gateKeeperName
	—		geoLocationName
	—		hlogStatus
	ignorePresentationIndicators		ignorePresentationIndicators
	—		index
	—		label
	—		line
	—		lineIdentifier
	—		lines
	loadInformation		—
	locationName		locationName
	—		maxNumCalls
	mediaResourceListName		mediaResourceListName
	mlppDomainId		mlppDomainId
	mlppIndicationStatus		—
	model		model
	—		monitoringCssName
	—		mtpPreferredCodec
	mtpRequired		mtpRequired
	name		name
	networkLocation		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	ownerUserId		—
	—		ownerUserName
	packetCaptureDuration		—
	packetCaptureMode		—
	—		partitionUsage
	—		presenceGroupName
	preemption		—
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		recordingFlag
	—		recordingProfileName
	—		redirectedNumber
	redirectInboundNumberIE		redirectInboundNumberIe
	redirectOutboundNumberIE		redirectOutboundNumberIe
	—		remoteDevice
	retryVideoCallAsAudio		retryVideoCallAsAudio
	—		routePartitionName
	signalingPort		signalingPort
	—		speedDial
	srtpAllowed		srtpAllowed
	subscribeCallingSearchSpaceName		subscribeCallingSearchSpaceName
	technologyPrefix		technologyPrefix
	traceFlag		traceFlag
	unattendedPort		unattendedPort
	—		useDevicePoolCgpnTransformCss
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		—
	versionStamp		—
	waitForFarEndH245TerminalSet		waitForFarEndH245TerminalSet
	zone		zone
H323Trunk		H323Trunk	
	aarNeighborhoodName		aarNeighborhoodName
	AllowH235PassThrough		allowH235PassThrough
	ASN1ROSEOIDEncoding		asn1RoseOidEncoding

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	calledNumberingPlan		calledNumberingPlan
	calledPartyIENumberType		calledPartyIeNumberType
	—		calledPartyInternationalPrefix
	—		calledPartyInternationalStripDigits
	—		calledPartyInternationalTransformationCssName
	—		calledPartyNationalPrefix
	—		calledPartyNationalStripDigits
	—		calledPartyNationalTransformationCssName
	—		calledPartySubscriberPrefix
	—		calledPartySubscriberStripDigits
	—		calledPartySubscriberTransformationCssName
	—		calledPartyUnknownPrefix
	—		calledPartyUnknownStripDigits
	—		calledPartyUnknownTransformationCssName
	callerIdDN		callerIdDn
	callingLineIdPresentation		callingLineIdPresentation
	callingNumberingPlan		callingNumberingPlan
	callingPartyIENumberType		callingPartyIeNumberType
	—		callingPartyInternationalStripDigits
	—		callingPartyInternationalTransformationCssName
	—		callingPartyNationalStripDigits
	—		callingPartyNationalTransformationCssName
	callingPartySelection		callingPartySelection
	—		callingPartySubscriberStripDigits
	—		callingPartySubscriberTransformationCssName
	—		callingPartyUnknownStripDigits
	—		callingPartyUnknownTransformationCssName
	callingSearchSpaceName		callingSearchSpaceName
	cdpnTransformationCSSName		cdpnTransformationCssName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	cgpnTransformationCSSName		cgpnTransformationCssName
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	cgpnTransformationUnknownCSSName		—
	class		class
	codecForOutboundFaststart		codecForOutboundFaststart
	commonDeviceConfigName		commonDeviceConfigName
	description		description
	devicePoolName		devicePoolName
	displayIEDelivery		displayIEDelivery
	enableInboundFaststart		enableInboundFaststart
	enableOutboundFaststart		enableOutboundFaststart
	—		gateKeeper
	gatekeeperInfo		gateKeeperInfo
	gatekeeperName		—
	—		geoLocationFilterName
	—		geoLocationName
	—		ictPassingPrecedenceLevelThroughUuie
	—		ictSecurityAccessLevel
	—		imeE164TransformationName
	internationalPrefix		internationalPrefix
	internationalStripDigits		—
	—		isSafEnabled
	licensedCapacity		—
	loadInformation		—
	locationName		locationName
	mediaResourceListName		mediaResourceListName
	mlppDomainId		mlppDomainId
	mlppIndicationStatus		mlppIndicationStatus
	model		model
	mtpRequired		mtpRequired

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	name		name
	nationalPrefix		nationalPrefix
	nationalStripDigits		—
	networkLocation		networkLocation
	packetCaptureDuration		packetCaptureDuration
	packetCaptureMode		packetCaptureMode
	pathReplacementSupport		pathReplacementSupport
	preemption		preemption
	prefixDN		prefixDn
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		pstnAccess
	QSIGVariant		qsigVariant
	redirectInboundNumberIE		redirectInboundNumberIe
	redirectOutboundNumberIE		redirectOutboundNumberIe
	—		remoteServerinfo
	retryVideoCallAsAudio		retryVideoCallAsAudio
	—		sendGeoLocation
	—		server1
	—		server2
	—		server3
	—		sigDigits
	—		signalingPort
	significantDigits		—
	srtpAllowed		srtpAllowed
	subscriberPrefix		subscriberPrefix
	subscriberStripDigits		—
	technologyPrefix		technologyPrefix
	terminal		—
	—		terminalType
	traceFlag		traceFlag
	—		transmitUtf8
	tunneledProtocol		tunneledProtocol
	tunneledProtocol		—
	unattendedPort		unattendedPort

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	unknownPrefix		unknownPrefix
	unknownStripDigits		—
	—		useDevicePoolCalledCssIntl
	—		useDevicePoolCalledCssNatl
	—		useDevicePoolCalledCssSubs
	—		useDevicePoolCalledCssUnkn
	useDevicePoolCdpnTransformCSS		useDevicePoolCdpnTransformCss
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	useDevicePoolCgpnTransformInternationalCSS		useDevicePoolCgpnTransformCssIntl
	useDevicePoolCgpnTransformNationalCSS		useDevicePoolCgpnTransformCssNatl
	useDevicePoolCgpnTransformSubscriberCSS		useDevicePoolCgpnTransformCssSubs
	useDevicePoolCgpnTransformUnknownCSS		useDevicePoolCgpnTransformCssUnkn
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		—
	versionStamp		—
	waitForFarEndH245TerminalSet		waitForFarEndH245TerminalSet
	zone		zone
HuntList		HuntList	
	calledPartyTransformationMask		—
	callingPartyPrefixDigits		—
	callingPartyTransformationMask		—
	callingSearchSpaceName		—
	callManagerGroupName		callManagerGroupName
	description		description
	dialPlanWizardGenId		—
	digitDiscardInstructionName		—
	lineGroupName		lineGroupName
	member		member
	members		members
	name		name
	prefixDigitsOut		—
	routeListEnabled		routeListEnabled
	selectionOrder		selectionOrder

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		voiceMailUsage
HuntPilot		HuntPilot	
	—		aarNeighborhoodName
	—		alertingName
	allowDeviceOverride		—
	—		asciiAlertingName
	blockEnable		blockEnable
	calledPartyNumberingPlan		calledPartyNumberingPlan
	calledPartyNumberType		calledPartyNumberType
	calledPartyTransformationMask		calledPartyTransformationMask
	callingLinePresentationBit		callingLinePresentationBit
	callingNamePresentationBit		callingNamePresentationBit
	callingPartyNumberingPlan		callingPartyNumberingPlan
	callingPartyNumberType		callingPartyNumberType
	callingPartyPrefixDigits		callingPartyPrefixDigits
	callingPartyTransformationMask		callingPartyTransformationMask
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	—		callingSearchSpaceName
	—		callPickupGroupName
	connectedLinePresentationBit		connectedLinePresentationBit
	connectedNamePresentationBit		connectedNamePresentationBit
	description		description
	destination		destination
	destination		destination
	destination		destination
	dialPlanName		dialPlanName
	dialPlanWizardGenId		—
	digitDiscardInstructionName		digitDiscardInstructionName
	—		e164Mask
	ForwardHuntBusy		forwardHuntBusy
	ForwardHuntNoAnswer		forwardHuntNoAnswer
	huntListName		huntListName
	maxHuntduration		maxHuntduration
	maxHuntduration		—
	messageWaiting		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	networkLocation		—
	ParkMonForwardNoRetrieveCSSName		—
	ParkMonForwardNoRetrieveDN		—
	—		parkMonForwardNoRetrieve
	pattern		pattern
	patternPrecedence		patternPrecedence
	patternUrgency		patternUrgency
	prefixDigitsOut		prefixDigitsOut
	provideOutsideDialTone		provideOutsideDialtone
	releaseCause		releaseClause
	routeFilterName		routeFilterName
	routePartitionName		routePartitionName
	supportOverlapSending		—
	usage		usage
	useCallingPartyPhoneMask		useCallingPartyPhoneMask
	usePersonalPreferences		usePersonalPreferences
	usePersonalPreferences		usePersonalPreferences
	—		usePersonalPreferences
IVRUserLocale		IvrUserLocale	
	orderIndex		orderIndex
	userLocale		userLocale
LicenseCapabilities		LicenseCapabilities	
	enableUPC		enableUpc
	—		enableUps
	userid		userid
Line		Line	
	aarDestinationMask		aarDestinationMask
	aarKeepCallHistory		aarKeepCallHistory
	aarNeighborhoodName		aarNeighborhoodName
	aarVoiceMailEnabled		aarVoiceMailEnabled
	alertingName		alertingName
	asciiAlertingName		asciiAlertingName
	autoAnswer		autoAnswer
	callForwardAll		callForwardAll
	callForwardAlternateParty		callForwardAlternateParty

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	callForwardBusy		callForwardBusy
	callForwardBusyInt		callForwardBusyInt
	callForwardNoAnswer		callForwardNoAnswer
	callForwardNoAnswerInt		callForwardNoAnswerInt
	callForwardNoCoverage		callForwardNoCoverage
	callForwardNoCoverageInt		callForwardNoCoverageInt
	callForwardNotRegistered		—
	callForwardNotRegisteredInt		callForwardNotRegisteredInt
	callForwardOnFailure		callForwardOnFailure
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		callingSearchSpaceName
	callingSearchSpaceName		—
	callPickupGroupName		callPickupGroupName
	cfaCSSClause		—
	cfaCSSPolicy		cfaCssPolicy
	—		defaultActivatedDeviceName
	description		description
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		destination
	destination		—
	destination		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	duration		duration
	duration		duration
	duration		—
	duration		—
	duration		—
	duration		—
	duration		—
	duration		—
	duration		—
	duration		—
	duration		—
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		forwardToVoiceMail
	forwardToVoiceMail		—
	hrDuration		hrDuration
	hrInterval		hrInterval
	networkHoldMOHAudioSourceId		networkHoldMohAudioSourceId
	ParkMonForwardNoRetrieveCSSName		parkMonForwardNoRetrieveCssName
	ParkMonForwardNoRetrieveDN		parkMonForwardNoRetrieveDn
	ParkMonForwardNoRetrieveIntCSSName		parkMonForwardNoRetrieveIntCssName
	ParkMonForwardNoRetrieveIntDN		parkMonForwardNoRetrieveIntDn
	ParkMonForwardNoRetrieveIntVMEnabled		parkMonForwardNoRetrieveIntVmEnabled
	ParkMonForwardNoRetrieveVMEnabled		parkMonForwardNoRetrieveVmEnabled
	ParkMonReversionTimer		parkMonReversionTimer
	partyEntranceTone		partyEntranceTone

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	pattern		pattern
	patternPrecedence		patternPrecedence
	presenceGroupName		presenceGroupName
	releaseCause		releaseClause
	routePartitionName		routePartitionName
	—		secondaryCallingSearchSpaceName
	shareLineAppearanceCSSName		shareLineAppearanceCssName
	usage		usage
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	voiceMailProfileName		voiceMailProfileName
LineGroup		LineGroup	
	—		directoryNumber
	distributionAlgorithm		distributionAlgorithm
	dnPattern		—
	dnPatternAndPartition		—
	huntAlgorithmBusy		huntAlgorithmBusy
	huntAlgorithmNoAnswer		huntAlgorithmNoAnswer
	huntAlgorithmNotAvailable		huntAlgorithmNotAvailable
	lineSelectionOrder		lineSelectionOrder
	member		member
	members		members
	name		name
	rnaReversionTimeOut		rnaReversionTimeOut
	routePartitionName		—
Location		Location	
	id		id
	kbits		kbits
	—		locationName
	name		name
	—		relatedLocation
	—		relatedLocations
	—		rsvpSetting
	videoKbits		videoKbits
MediaResourceGroup		MediaResourceGroup	
	description		description
	deviceName		deviceName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	deviceName		—
	deviceName		—
	member		member
	member		—
	member		—
	members		members
	multicast		multicast
	name		name
MediaResourceList		MediaResourceList	
	clause		clause
	mediaResourceGroupName		mediaResourceGroupName
	mediaResourceGroupName		—
	member		—
	member		member
	members		members
	name		name
	—		order
MeetMe		MeetMe	
	description		description
	minimumSecurityLevel		minimumSecurityLevel
	pattern		pattern
	routePartitionName		routePartitionName
	usage		usage
MobileSmartClientProfile		MobileSmartClientProfile	
	enableCFAUri		enableCFAUri
	enableSNRUri		enableSnrUri
	handOffUri		handOffUri
	mobileSmartClient		mobileSmartClient
	name		name
MobileVoiceAccess		MobileVoiceAccess	
	—		locale
	—		locales
	—		orderIndex
	pattern		pattern
	routePartitionName		routePartitionName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		userLocale
MOHServer		MohServer	
	—		audioSource
	—		audioSources
	—		baseMulticastIpaddress
	—		baseMulticastPort
	—		description
	devicePoolName		devicePoolName
	fixedAudioSourceDevice		fixedAudioSourceDevice
	isMultiCastEnabled		isMultiCastEnabled
	locationName		locationName
	—		maxHops
	maxMulticastConnections		maxMulticastConnections
	maxUnicastConnections		maxUnicastConnections
	—		multicastIncrementOnIp
	name		name
	—		processNodeName
	runFlag		runFlag
	—		sourceId
	—		useTrustedRelayPoint
Phone		Phone	
	aarDestinationMask		—
	aarKeepCallHistory		—
	aarNeighborhoodName		—
	aarNeighborhoodName		aarNeighborhoodName
	aarVoiceMailEnabled		—
	addOnModule		addOnModule
	addOnModules		addOnModules
	—		addressMode
	alertingName		—
	—		allowAutoConfig
	allowCtiControlFlag		allowCtiControlFlag
	alwaysUsePrimeLine		alwaysUsePrimeLine
	—		alwaysUsePrimeLine
	alwaysUsePrimeLineforVoiceMessage		alwaysUsePrimeLineForVoiceMessage

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		alwaysUsePrimeLineForVoiceMessage
	asciiLabel		asciiLabel
	—		asciiLabel
	—		asciiLabel
	—		asciiLabel
	—		associatedBlfSdFeatures
	—		associatedEndusers
	associatedPC		—
	—		audibleMwi
	authenticationMode		authenticationMode
	authenticationString		authenticationString
	authenticationURL		authenticationUrl
	autoAnswer		
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	automatedAlternateRoutingCSSName		—
	blfDest		blfDest
	—		blfDest
	—		blfDirectedCallPark
	—		blfDirectedCallParks
	blfDirn		blfDirn
	—		blfDirn
	builtInBridgeStatus		builtInBridgeStatus
	busyLampField		busyLampField
	—		busyLampField
	busyLampFields		busyLampFields
	—		busyLampFields
	busyTrigger		busyTrigger
	callerName		callerName
	callerNumber		callerNumber
	callForwardAll		—
	callForwardAlternateParty		—
	callForwardBusy		—
	callForwardBusyInt		—
	callForwardNoAnswer		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	callForwardNoAnswerInt		—
	callForwardNoCoverage		—
	callForwardNoCoverageInt		—
	callForwardNotRegistered		—
	callForwardNotRegisteredInt		—
	callForwardOnFailure		—
	callInfoDisplay		callInfoDisplay
	callInfoPrivacyStatus		callInfoPrivacyStatus
	—		callInfoPrivacyStatus
	callingSearchSpaceName		callingSearchSpaceName
	callPickupGroupName		—
	certificateOperation		certificateOperation
	certificateStatus		certificateStatus
	cfaCSSClause		—
	cfaCSSPolicy		—
	cgpnTransformationCSSName		cgpnTransformationCssName
	class		class
	—		class
	commonDeviceConfigName		commonDeviceConfigName
	—		commonPhoneConfigName
	consecutiveRingSetting		consecutiveRingSetting
	—		currentConfig
	—		currentProfileName
	—		defaultProfileName
	description		description
	description		—
	deviceMobilityMode		deviceMobilityMode
	—		deviceName
	devicePoolName		devicePoolName
	—		deviceProtocol
	dialedNumber		dialedNumber
	dialPlanWizardId		dialPlanWizardId
	—		dialRulesName
	—		digestUser
	—		directedCallParkDnAndPartition
	—		directedCallParkId

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		directoryNumber
	directoryURL		directoryUrl
	dirn		dirn
	dirn		dirn
	display		display
	—		displayAscii
	dndOption		dndOption
	—		dndOption
	dndRingSetting		dndRingSetting
	—		dndRingSetting
	—		dndStatus
	—		dndStatus
	—		dnPattern
	e164Mask		e164Mask
	—		emccCallingSearchSpaceName
	enableExtensionMobility		enableExtensionMobility
	—		enduser
	—		feature
	—		geoLocationFilterName
	—		geolocationInfo
	—		geoLocationName
	hlogStatus		hlogStatus
	—		hotlineDevice
	hrDuration		—
	hrInterval		—
	idleTimeout		idleTimeout
	idleURL		idleUrl
	ignorePresentationIndicators		ignorePresentationIndicators
	—		ignorePresentationIndicators
	—		index
	—		index
	—		index
	—		index
	—		index
	—		index
	informationURL		informationUrl

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	isActive		isActive
	—		isDualMode
	isProtected		isProtected
	joinAcrossLines		joinAcrossLines
	—		joinAcrossLines
	keySize		keySize
	label		label
	label		label
	label		label
	—		label
	—		label
	—		labelAscii
	line		line
	—		lineIdentifier
	lines		lines
	loadInformation		loadInformation
	loadInformation		loadInformation
	locationName		locationName
	—		loginDuration
	—		loginTime
	—		loginUserId
	maxNumCalls		maxNumCalls
	mediaResourceListName		mediaResourceListName
	messagesURL		messagesUrl
	mlppDomainId		mlppDomainId
	—		mlppDomainId
	mlppIndicationStatus		mlppIndicationStatus
	—		mlppIndicationStatus
	mobileSmartClientProfileName		mobileSmartClientProfileName
	—		mobilityUserIdName
	model		model
	model		model
	—		model
	—		monitoringCssName
	—		mtpPreferedCodec
	—		mtpRequired

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	mwlPolicy		mwlPolicy
	name		name
	—		name
	networkHoldMOHAudioSourceId		networkHoldMohAudioSourceId
	networkHoldMOHAudioSourceId		—
	networkLocale		networkLocale
	networkLocation		networkLocation
	numberOfButtons		numberOfButtons
	—		outboundCallRollover
	ownerUserId		—
	—		ownerUserName
	packetCaptureDuration		packetCaptureDuration
	packetCaptureMode		packetCaptureMode
	—		partitionUsage
	pattern		—
	patternPrecedence		—
	phoneLoadName		—
	—		phoneService
	—		phoneServiceCategory
	phoneServiceDisplay		phoneServiceDisplay
	phoneSuite		phoneSuite
	phoneTemplateName		phoneTemplateName
	—		phoneTemplateName
	preemption		preemption
	—		preemption
	presenceGroupName		presenceGroupName
	presenceGroupName		—
	—		primaryPhoneName
	—		priority
	product		product
	—		product
	protocol		protocol
	protocolSide		protocolSide
	proxyServerURL		proxyServerUrl
	—		recordingFlag
	—		recordingProfileName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	redirectedNumber		redirectedNumber
	releaseCause		—
	remoteDevice		remoteDevice
	—		remoteSipSrstIp
	—		remoteSipSrstPort
	—		remoteSrstIp
	—		remoteSrstOption
	—		remoteSrstPort
	requireDTMFReception		requireDtmfReception
	rerouteCallingSearchSpace		rerouteCallingSearchSpaceName
	retryVideoCallAsAudio		retryVideoCallAsAudio
	RFC2833Disabled		rfc2833Disabled
	ringSetting		ringSetting
	—		ringSettingActivePickupAlert
	ringSettingBusyBLFAudibleAlert		ringSettingBusyBlfAudibleAlert
	ringSettingIdleBLFAudibleAlert		ringSettingIdleBlfAudibleAlert
	—		ringSettingIdlePickupAlert
	—		roamingDevicePoolName
	—		routePartition
	—		routePartition
	routePartitionName		routePartitionName
	—		routePartitionName
	—		secureAuthenticationUrl
	—		secureDirectoryUrl
	—		secureIdleUrl
	—		secureInformationUrl
	—		secureMessageUrl
	—		secureServicesUrl
	—		securityProfileName
	—		sendGeoLocation
	—		serviceNameAscii
	—		services
	servicesURL		servicesUrl
	shareLineAppearanceCSSName		—
	singleButtonBarge		singleButtonBarge
	—		singleButtonBarge

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		sipProfileName
	softkeyTemplateName		softkeyTemplateName
	—		softkeyTemplateName
	speeddial		speeddial
	—		speedDial
	speeddials		speeddials
	—		sshPwd
	—		sshUserId
	subscribeCallingSearchSpace		—
	—		subscribeCallingSearchSpaceName
	—		telecasterServiceName
	traceFlag		traceFlag
	unattendedPort		unattendedPort
	upgradeFinishTime		upgradeFinishTime
	—		url
	—		urlButtonIndex
	—		urlLabel
	—		urlLabelAscii
	usage		—
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userLocale		userlocale
	—		useTrustedRelayPoint
	—		vendor
	vendorConfig		vendorConfig
	—		version
	versionStamp		versionStamp
	voiceMailProfileName		
PhoneTemplate		PhoneButtonTemplate	
	basePhoneTemplateName		basePhoneTemplateName
	button		button
	—		buttonNumber
	buttons		buttons
	feature		feature

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	isUserModifiable		isUserModifiable
	label		label
	name		name
PhysicalLocation		PhysicalLocation	
	—		description
	name		name
ProcessNode		ProcessNode	
	description		description
	IPv6Name		ipv6Name
	mac		mac
	name		name
			nodeUsage
ProcessNodeService		ProcessNodeService	
	deviceNameMonitorFlag		—
	deviceTypeMonitorFlag		—
	enable		enable
	fileTraceFlag		—
	includeNonDeviceTraces		—
	—		isActive
	—		maxFileSize
	numFiles		numFiles
	numLines		—
	numMinutes		—
	outputDebugStringFlag		—
	—		processNodeName
	serverName		—
	service		service
	traceLevel		traceLevel
	userCategories		userCategories
	useXML		—
	deviceNameMonitorFlag		—
RecordingProfile		RecordingProfile	
	name		name
	recordingCSSName		recordingCssName
	recorderDestination		recorderDestination
Region		Region	

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	defaultCodec		defaultCodec
	name		name
	—		bandwidth
	—		lossyNetwork
	—		regionName
	—		relatedRegion
	—		relatedRegions
	—		videoBandwidth
RegionMatrix		RegionMatrix	
	bandwidth		bandwidth
	lossyNetwork		lossyNetwork
	—		regionAName
	—		regionBName
	videoBandwidth		videoBandwidth
RemoteDestination		RemoteDestination	
	answerTooLateTimer		answerTooLateTimer
	answerTooSoonTimer		answerTooSoonTimer
	delayBeforeRingingCell		delayBeforeRingingCell
	destination		destination
	—		dualModeDeviceName
	enableMobileConnect		enableMobileConnect
	—		index
	isMobilePhone		isMobilePhone
	—		lineAssociation
	—		lineAssociations
	mobileSmartClientName		mobileSmartClientName
	name		name
	—		pattern
	remoteDestinationProfileName		remoteDestinationProfileName
	—		routePartitionName
	timeZone		timeZone
	todAccessName		todAccessName
	—		uuid
RemoteDestinationProfile		RemoteDestinationProfile	
	—		asciiLabel

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		associatedEndusers
	—		audibleMwi
	—		busyTrigger
	—		callerName
	—		callerNumber
	—		callInfoDisplay
	callInfoPrivacyStatus		callInfoPrivacyStatus
	callingSearchSpaceName		callingSearchSpaceName
	cgpnTransformationCSSName		cgpnTransformationCssName
	class		class
	—		consecutiveRingSetting
	description		description
	devicePoolName		devicePoolName
	—		dialedNumber
	—		dialPlanWizardId
	—		directoryNumber
	—		dirn
	—		display
	—		displayAscii
	dndOption		dndOption
	dndRingSetting		—
	dndStatus		dndStatus
	—		e164Mask
	—		enduser
	ignorePresentationIndicators		ignorePresentationIndicators
	—		index
	—		label
	—		line
	—		lineIdentifier
	—		lines
	—		maxNumCalls
	mobileSmartClientProfileName		mobileSmartClientProfileName
	model		model
	—		monitoringCssName
	—		mwlPolicy
	name		name

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	networkHoldMOHAudioSourceId		networkHoldMohAudioSourceId
	—		partitionUsage
	—		primaryPhoneName
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		recordingFlag
	—		recordingProfileName
	—		redirectedNumber
	rerouteCallingSearchSpace		—
	—		rerouteCallingSearchSpaceName
	—		ringSetting
	—		ringSettingActivePickupAlert
	—		ringSettingIdlePickupAlert
	—		routePartitionName
	—		speedDial
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	userHoldMOHAudioSourceId		userHoldMohAudioSourceId
	userId		userId
			userLocale
ResourcePriorityNamespace		ResourcePriorityNamespace	
	—		description
	—		isDefault
	namespace		namespace
ResourcePriorityNamespaceList		ResourcePriorityNamespaceList	
	—		description
	—		index
	member		member
	member		—
	members		members
	name		name
	resourcePriorityNamespaceName		resourcePriorityNamespaceName
	resourcePriorityNamespaceName		—
RouteFilter		RouteFilter	
	clause		clause

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	dialPlanName		dialPlanName
	dialPlanTagName		dialPlanTagName
	dialPlanTagName		—
	dialPlanWizardGenId		dialPlanWizardGenId
	digits		digits
	digits		—
	member		member
	member		—
	members		members
	name		name
	operator		operator
	operator		—
	—		priority
RouteGroup		RouteGroup	
	deviceName		deviceName
	deviceName		
	deviceSelectionOrder		deviceSelectionOrder
	deviceSelectionOrder		
	dialPlanWizardGenId		dialPlanWizardGenId
	dialPlanWizardGenId		dialPlanWizardGenId
	dialPlanWizardGenId		—
	—		distributionAlgorithm
	member		member
	member		—
	members		members
	name		name
	—		port
RouteList		RouteList	
	calledPartyNumberingPlan		calledPartyNumberingPlan
	calledPartyNumberType		calledPartyNumberType
	calledPartyTransformationMask		calledPartyTransformationMask
	callingPartyNumberingPlan		callingPartyNumberingPlan
	callingPartyNumberType		callingPartyNumberType
	callingPartyPrefixDigits		callingPartyPrefixDigits
	callingPartyTransformationMask		callingPartyTransformationMask
	callingSearchSpaceName		—

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	callManagerGroupName		callManagerGroupName
	description		description
	dialPlanWizardGenId		dialPlanWizardGenId
	digitDiscardInstructionName		digitDiscardInstructionName
	member		member
	members		members
	name		name
	prefixDigitsOut		prefixDigitsOut
	routeGroupName		routeGroupName
	routeListEnabled		routeListEnabled
	selectionOrder		selectionOrder
	useFullyQualifiedCallingPartyNumber		useFullyQualifiedCallingPartyNumber
RoutePartition		RoutePattern	
	allowDeviceOverride		allowDeviceOverride
	authorizationCodeRequired		authorizationCodeRequired
	authorizationLevelRequired		authorizationLevelRequired
	blockEnable		blockEnable
	calledPartyNumberingPlan		calledPartyNumberingPlan
	calledPartyNumberType		calledPartyNumberType
	calledPartyTransformationMask		calledPartyTransformationMask
	callingLinePresentationBit		callingLinePresentationBit
	callingNamePresentationBit		callingNamePresentationBit
	callingPartyNumberingPlan		callingPartyNumberingPlan
	callingPartyNumberType		callingPartyNumberType
	callingPartyPrefixDigits		callingPartyPrefixDigits
	callingPartyTransformationMask		callingPartyTransformationMask
	cic		cic
	clientCodeRequired		clientCodeRequired
	connectedLinePresentationBit		connectedLinePresentationBit
	—		connectedLinePresentationBit
	connectedNamePresentationBit		connectedNamePresentationBit
	—		connectedNamePresentationBit
	description		description
	destination		destination
	dialPlanName		dialPlanName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	dialPlanWizardGenId		dialPlanWizardGenId
	digitDiscardInstructionName		digitDiscardInstructionName
	—		gatewayName
	isdnNFSInfoElement		isdnNsfInfoElement
	messageWaiting		—
	networkLocation		networkLocation
	—		networkService
	networkServiceProtocol		networkServiceProtocol
	—		paramValue
	pattern		pattern
	patternPrecedence		patternPrecedence
	patternUrgency		patternUrgency
	prefixDigitsOut		prefixDigitsOut
	provideOutsideDialtone		provideOutsideDialtone
	releaseCause		releaseClause
	resourcePriorityNameSpaceName		resourcePriorityNamespaceName
	—		routeClass
	routeFilterName		routeFilterName
	routeListName		routeListName
	routePartitionName		routePartitionName
	supportOverlapSending		supportOverlapSending
	usage		usage
	useCallingPartyPhoneMask		useCallingPartyPhoneMask
	withTag		withTag
	withValueClause		withValueClause
RoutePattern		RoutePartition	
	description		description
	dialPlanWizardGenId		dialPlanWizardGenId
	name		name
	partitionUsage		partitionUsage
	timeScheduleName		timeScheduleIdName
	timeZone		timeZone
	useOriginatingDeviceTimeZone		useOriginatingDeviceTimeZone
ServiceParameter		ServiceParameter	
	name		name
	newValue		

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	processNodeName		processNodeName
	service		service
	—		value
	—		valueType
SIPProfile		SipProfile	
	abbreviatedDialURI		abbreviatedDialUri
	anonymousCallBlock		anonymousCallBlock
	callerIdBlock		callerIdBlock
	callForwardURI		callForwardUri
	callHoldRingback		callHoldRingback
	callpickupGroupURI		callpickupGroupUri
	callpickupListURI		callpickupListUri
	callpickupURI		callpickupUri
	callStats		callStats
	confJoinEnable		confJointEnable
	defaultTelephonyEventPayloadType		defaultTelephonyEventPayloadType
	description		description
	dndControl		dndControl
	dtmfDbLevel		dtmfDbLevel
	enableVAD		enableVad
	maxRedirects		maxRedirects
	meetmeServiceURI		meetmeServiceUrl
	name		name
	redirectByApplication		redirectByApplication
	rerouteIncomingRequest		rerouteIncomingRequest
	resourcePriorityNamespaceListName		resourcePriorityNamespaceListName
	retryInvite		retryInvite
	retryNotInvite		retryNotInvite
	rfc2543Hold		rfc2543Hold
	ringing180		ringing180
	semiAttendedTransfer		semiAttendedTransfer
	startMediaPort		startMediaPort
	stopMediaPort		stopMediaPort
	stutterMsgWaiting		stutterMsgWaiting
	t38Invite		t38Invite

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	telnetLevel		telnetLevel
	timerInvite		timerInvite
	timerKeepAlive		timerKeepAlive
	timerOffhookToFirstDigit		timerOffHookToFirstDigit
	timerRegister		timerRegister
	timerRegisterDelta		timerRegisterDelta
	timerSubscribe		timerSubscribe
	timerSubscribeDelta		timerSubscribeDelta
	timerT1		timerT1
	timerT2		timerT2
	userInfo		userInfo
	—		enableAnatForEarlyOfferCalls
	—		fallbackToLocalRsvp
	—		gClear
	—		rsvpOverSip
	—		sipRe11XxEnabled
SIPRealm		SipRealm	
	digestCredentials		digestCredentials
	realm		realm
	userid		userid
SIPRoutePattern		SipRoutePattern	
	blockEnable		blockEnable
	callingLinePresentationBit		callingLinePresentationBit
	callingNamePresentationBit		callingNamePresentationBit
	callingPartyPrefixDigits		callingPartyPrefixDigits
	callingPartyTransformationMask		callingPartyTransformationMask
	connectedLinePresentationBit		connectedLinePresentationBit
	connectedNamePresentationBit		connectedNamePresentationBit
	description		description
	destination		—
	dnOrPatternIPv6		dnOrPatternIpv6
	—		domainRoutingCssName
	pattern		pattern
	—		routeOnUserPart
	routePartitionName		routePartitionName
	sipTrunkName		sipTrunkName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	usage		usage
	—		useCallerCss
	useCallingPartyPhoneMask		useCallingPartyPhoneMask
SIPTrunk		SipTrunk	
	—		aarNeighborhoodName
	acceptInboundRDNIS		acceptInboundRdnis
	acceptOutboundRDNIS		acceptOutboundRdnis
	automatedAlternateRoutingCSS		automatedAlternateRoutingCssName
	callerIdDN		callerIdDn
	callerName		callerName
	callingLineIdPresentation		callingLineIdPresentation
	callingname		callingname
	callingPartySelection		callingPartySelection
	callingSearchSpaceName		callingSearchSpaceName
	cdpnTransformationCSSName		cdpnTransformationCssName
	cgpnTransformationCSSName		cgpnTransformationCssName
	cgpnTransformationUnknownCSSName		—
	class		class
	—		cntdPnTransformationCssName
	—		commonDeviceConfigName
	connectedNamePresentation		connectedNamePresentation
	connectedPartyIdPresentation		connectedPartyIdPresentation
	description		description
	destAddrIsSRV		destAddrIsSrv
	destinationaddress		destinationAddress
	destinationAddressIPv6		destinationAddressIpv6
	destinationport		destinationport
	devicePoolName		devicePoolName
	—		dtmfSignalingMethod
	—		geoLocationFilterName
	—		geoLocationName
	—		imeE164TransformationName
	incomingport		—
	isPaiEnabled		isPaiEnabled
	isRpidEnabled		isRpidEnabled

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	licensedCapacity		—
	loadInformation		loadInformation
	locationName		locationName
	mediaResourceListName		mediaResourceListName
	mlppDomainId		mlppDomainId
	mlppIndicationStatus		mlppIndicationStatus
	model		model
	mtpRequired		mtpRequired
	name		name
	Name		—
	—		networkHoldMohAudioSourceId
	networkLocation		networkLocation
	OutgoingTransportType		—
	—		packetCaptureDuration
	—		packetCaptureMode
	preemption		preemption
	prefixDN		prefixDn
	—		presenceGroupName
	product		product
	protocol		protocol
	protocolSide		protocolSide
	—		pstnAccess
	referCallingSearchSpaceName		referCallingSearchSpaceName
	rerouteCallingSearchSpaceName		rerouteCallingSearchSpaceName
	retryVideoCallAsAudio		retryVideoCallAsAudio
	—		routeClassSignalling
	securityProfileName		securityProfileName
	—		sendGeoLocation
	sigDigits		sigDigits
	sipAssertedType		sipAssertedType
	sipPrivacy		sipPrivacy
	sipProfileName		sipProfileName
	—		sipTrunkType
	srtpAllowed		srtpAllowed
	srtpFallbackAllowed		srtpFallbackAllowed
	subscribeCallingSearchSpaceName		subscribeCallingSearchSpaceName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	tkSipCodec		tkSipCodec
	traceFlag		traceFlag
	—		transmitUtf8
	—		unattendedPort
	—		unknownPrefix
	useDevicePoolCdpnTransformCSS		useDevicePoolCdpnTransformCss
	useDevicePoolCgpnTransformCSS		useDevicePoolCgpnTransformCss
	useDevicePoolCgpnTransformUnknownCSS		—
	—		useDevicePoolCntdPnTransformationCss
	—		useImePublicIpPort
	—		userHoldMohAudioSourceId
	useTrustedRelayPoint		useTrustedRelayPoint
	vendorConfig		
	versionStamp		versionStamp
SIPTrunkSecurityProfile		SipTrunkSecurityProfile	
	acceptOutOfDialogRefer		acceptOutOfDialogRefer
	acceptPresenceSubscription		acceptPresenceSubscription
	acceptUnsolicitedNotification		acceptUnsolicitedNotification
	allowReplaceHeader		allowReplaceHeader
	applLevelAuthentication		applLevelAuthentication
	description		description
	digestAuthentication		digestAuthentication
	incomingPort		incomingPort
	incomingTransport		incomingTransport
	name		name
	noncePolicyTime		noncePolicyTime
	outgoingTransport		outgoingTransport
	securityMode		securityMode
	transmitSecurityStatus		transmitSecurityStatus
	x509SubjectName		x509SubjectName
SoftKeyTemplate		SoftKeyTemplate	
	application		application
	applications		applications
	baseSoftkeyTemplateName		baseSoftkeyTemplateName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	description		description
	isDefault		isDefault
	name		name
TimePeriod		TimePeriod	
	dayOfMonth		dayOfMonth
	dayOfMonthEnd		dayOfMonthEnd
	description		description
	endDay		endDay
	endTime		endTime
	isPublished		isPublished
	monthOfYear		monthOfYear
	monthOfYearEnd		monthOfYearEnd
	name		name
	startDay		startDay
	startTime		startTime
			todOwnerIdName
TimeSchedule		TimeSchedule	
	isPublished		description
	member		isPublished
	members		member
	name		members
	timePeriodName		name
	timeScheduleCategory		timeScheduleCategory
	—		todOwnerIdName
TODAccess		TodAccess	
	name		name
	ownerIdName		ownerIdName
	—		associatedRemoteDestination
	—		description
	—		member
	—		members
	—		remoteDestination
Transcoder		Transcoder	
	commonDeviceConfigName		commonDeviceConfigName
	description		description
	devicePoolName		devicePoolName

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	isTrustedRelayPoint		isTrustedRelayPoint
	loadInformation		loadInformation
	maximumCapacity		maximumCapacity
	name		name
	product		product
	subUnit		subUnit
	vendorConfig		vendorConfig
TransPattern		TransPattern	
	allowDeviceOverride		—
	blockEnable		blockEnable
	calledPartyNumberingPlan		calledPartyNumberingPlan
	calledPartyNumberType		calledPartyNumberType
	calledPartyTransformationMask		calledPartyTransformationMask
	callingLinePresentationBit		callingLinePresentationBit
	callingNamePresentationBit		callingNamePresentationBit
	callingPartyNumberingPlan		callingPartyNumberingPlan
	callingPartyNumberType		callingPartyNumberType
	callingPartyPrefixDigits		callingPartyPrefixDigits
	callingPartyTransformationMask		callingPartyTransformationMask
	callingSearchSpaceName		callingSearchSpaceName
	connectedLinePresentationBit		callInterceptProfileName
	connectedNamePresentationBit		connectedLinePresentationBit
	description		connectedNamePresentationBit
	dialPlanName		description
	dialPlanWizardGenId		—
	digitDiscardInstructionName		digitDiscardInstructionName
	messageWaiting		—
	networkLocation		—
	pattern		pattern
	patternPrecedence		patternPrecedence
	patternUrgency		patternUrgency
	prefixDigitsOut		prefixDigitsOut
	provideOutsideDialtone		provideOutsideDialtone
	releaseCause		releaseClause
	resourcePriorityNameSpaceName		resourcePriorityNamespaceName
	—		routeClass

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	routeFilterName		routeFilterName
	—		routeNextHopByCgpn
	routePartitionName		routePartitionName
	usage		usage
	useCallingPartyPhoneMask		useCallingPartyPhoneMask
User		User	
	—		associatedCapfProfiles
	associatedDevices		associatedDevices
	—		associatedGroups
	associatedPC		associatedPc
	—		associatedTodAccess
	—		capfProfileInstanceId
	—		ctiControlledDeviceProfiles
	—		defaultProfile
	department		department
	device		device
	digestCredentials		digestCredentials
	enableCTI		enableCti
	—		enableEmcc
	enableMobileVoiceAccess		enableMobileVoiceAccess
	enableMobility		enableMobility
	extension		—
	firstname		firstName
	lastname		lastName
	—		mailid
	manager		manager
	MaxDeskPickupWaitTime		maxDeskPickupWaitTime
	—		middleName
	—		name
	password		password
	pattern		pattern
	phoneProfiles		phoneProfiles
	pin		pin
	—		pinCredDoesNotExpire
	—		pinCredentials
	—		pinCredLockedByAdministrator

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	—		pinCredPolicyName
	—		pinCredTimeAdminLockout
	—		pinCredTimeChanged
	—		pinCredUserCantChange
	—		pinCredUserMustChange
	—		presenceGroupName
	—		primaryDevice
	primaryExtension		primaryExtension
	profileName		profileName
	—		profileName
	—		pwdCredDoesNotExpire
	—		pwdCredLockedByAdministrator
	—		pwdCredTimeAdminLockout
	—		pwdCredTimeChanged
	—		pwdCredUserCantChange
	—		pwdCredUserMustChange
	remoteDestinationLimit		—
	routePartitionName		routePartitionName
	—		status
	—		subscribeCallingSearchSpaceName
	telephoneNumber		telephoneNumber
	—		todAccess
	—		userGroup
	userid		userid
	userLocale		userLocale
	—		userRole
	—		userRoles
UserGroup		UserGroup	
	member		member
	members		members
	name		name
	userId		userId
VG224		Vg224	
			beginPort
	callManagerGroupName		callManagerGroupName
	description		description

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	domainName		domainName
	—		index
	—		index
	product		product
	product		product
	—		product
	protocol		protocol
	subunit		subunit
	subunits		subunits
	unit		unit
	units		units
	—		vendorConfig
	—		versionStamp
VoiceMailPilot		VoiceMailPilot	
	CSSName		cssName
	description		description
	dirn		dirn
	isDefault		isDefault
VoiceMailPort		VoiceMailPort	
	—		aarNeighborhoodName
	automatedAlternateRoutingCSSName		automatedAlternateRoutingCssName
	—		callerIdDisplay
	—		callerIdDisplayAscii
	callingSearchSpaceName		callingSearchSpaceName
	—		class
	commonDeviceConfigName		commonDeviceConfigName
	description		description
	devicePoolName		devicePoolName
	—		dnCallingSearchSpace
	—		dnPattern
	—		externalMask
	—		geoLocationName
	loadInformation		—
	locationName		locationName
	—		model

Table 2-4 Changed Operations in Unified CM 8.0

API in 7.1	Tags in 7.1	API in 8.0(1)	Tags in 8.0(1)
	name		name
	—		preemption
	—		product
	—		protocol
	—		protocolSide
	—		routePartition
	—		securityProfileName
	traceFlag		
	useTrustedRelayPoint		useTrustedRelayPoint
VoiceMailProfile		VoiceMailProfile	
	—		cssName
	description		description
	—		dirn
	isDefault		isDefault
	name		name
	voiceMailboxMask		voiceMailboxMask
	—		voiceMailPilot

Schema and Other Changes

The following changes are made in the AXL schema in release 8.0(1):

- Single file for schema called AXLSoap.xsd and it is a combination of axl.xsd and axlsoap.xsd.
- Changes in API and tag names in several APIs. These changes ensures consistency in naming APIs and their tags across all the AXL APIs. The following examples ilustrates the changes made:
  - In addAARGroup handler, AARGroup element is changed to aarGroup. See Figure 2-1 and Figure 2-2.

Figure 2-1 addAARGroup element in Unified CM Release 7.1(2)

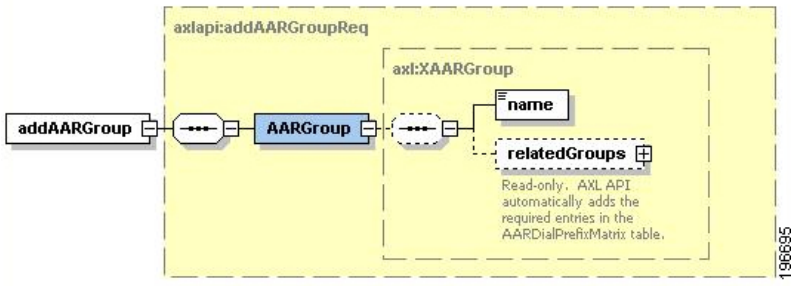
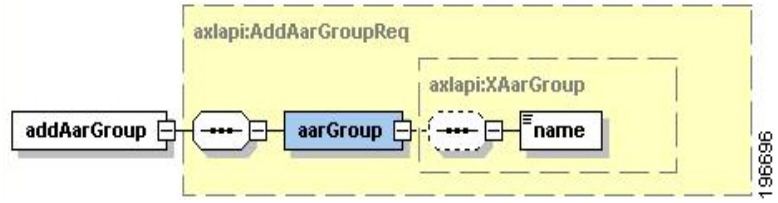


Figure 2-2 addAarGroup element in Unified CM Release 8.0(1)



– In addAppUser handler the newApplicationUser element is replaced with appUser. See Figure 2-3 and Figure 2-4.

Figure 2-3 newAppicationUser element in Unified CM 7.1(2)

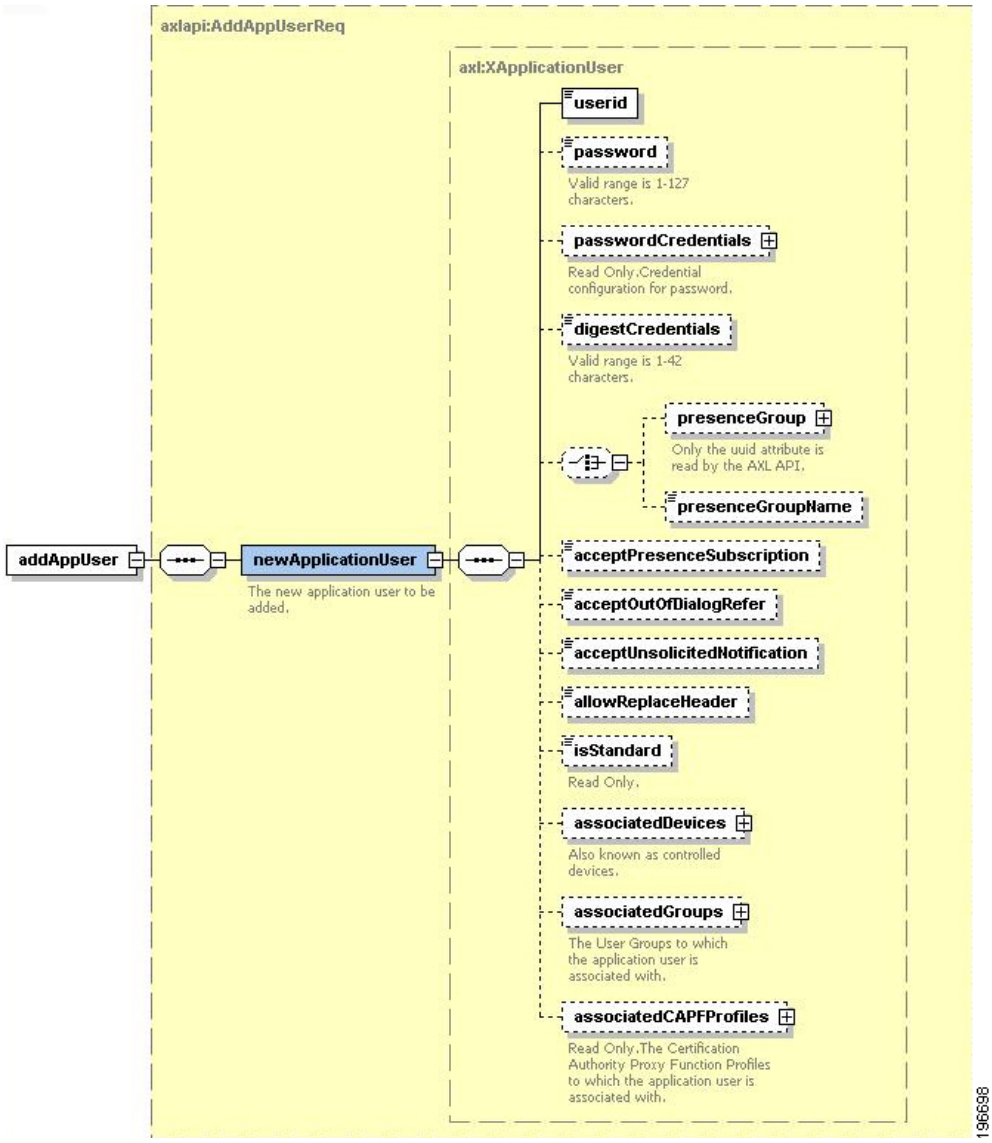
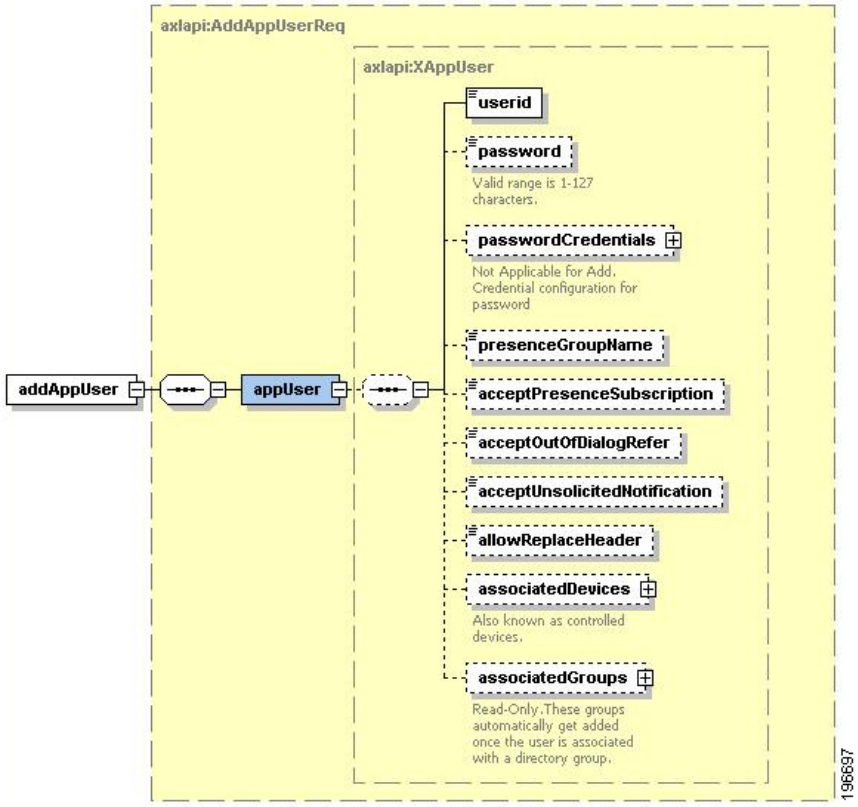


Figure 2-4 appUser element in Unified CM 8.0(1)



– In addCalledPartyTransformationPattern handler the newPattern element is replaced with calledPartyTransformationPattern. See Figure 2-5 and Figure 2-6.

Figure 2-5 newPattern element in Unified CM 7.1(2)

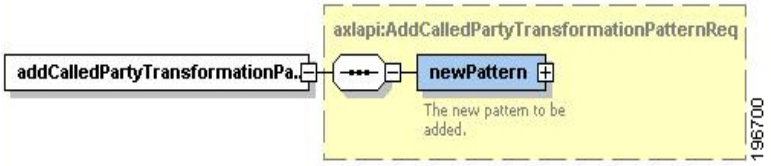
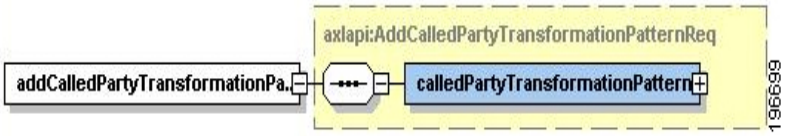


Figure 2-6 calledPartyTransformationPattern element in Unified CM 8.0(1)



– In addRecordingProfile handler the recordingCSSName element is replaced with recordingCssName. See Figure 2-7 and Figure 2-8.

Figure 2-7 recordingCSSName element in Unified CM 7.1(2)

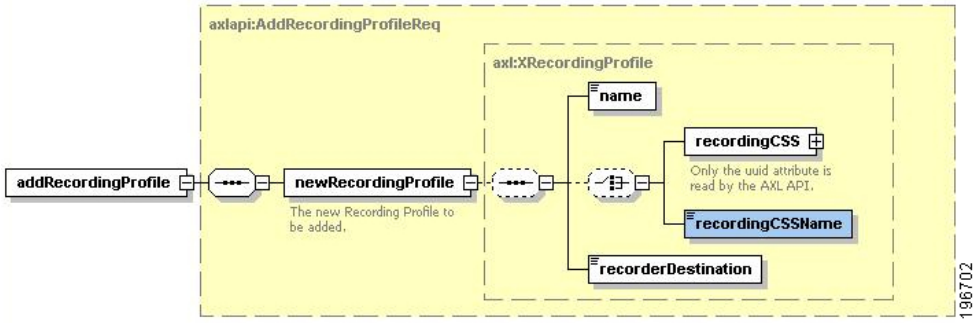
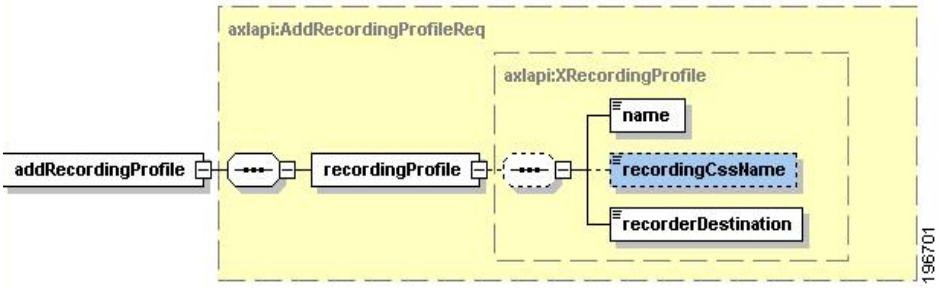


Figure 2-8 recordingCssName element in Unified CM 8.0(1)



- The Get and List requests have a new optional tag called `returnedTags`. This tag has all the tags of that API as optional subtags. If any of the tags are sent in the request, only those tags are returned in the response. Also, there is a new `searchCriteria` tag in which you can specify the parameters for the search. The `searchCriteria` tag is present only for List APIs and is not available in Get APIs. Figure 2-9 illustrates the implementation for Get and List methods in AXL from Unified CM 8.0.

Figure 2-9 Optional Tags—returnedTags in Get Operation

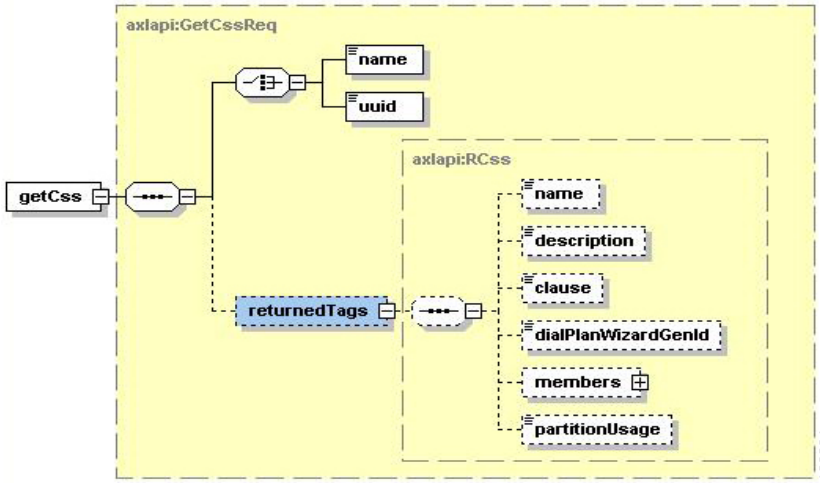
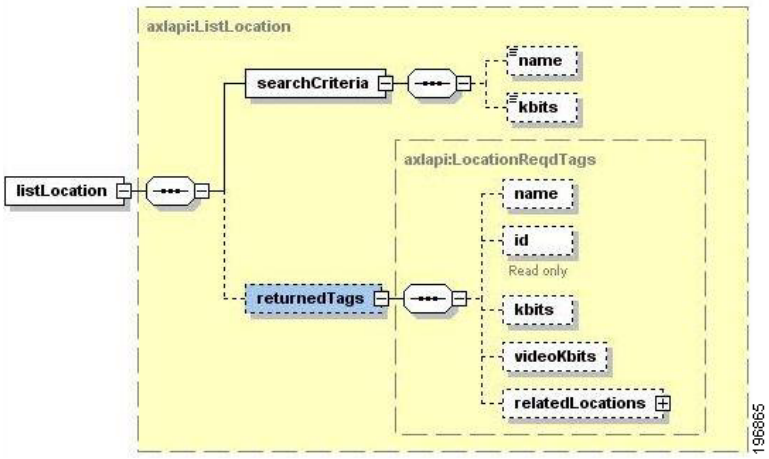


Figure 2-10 Optional Tags—searchCriteria and returnedTags in List Operation



- There are no attributes listed in any tag. The only exception is the uuid attribute. The attributes found in various tags, in earlier Unified CM releases, have been replaced with tags. For example, index attributes present in line, gateway, and member tags are removed and index tags have been added.
- A uuid attribute added to the `name` tag. There is no `uuid` tag in any of the operations.
- In Unified CM 8.0(1) the AXL returns **500 Internal Server Error** response status code for indicating internal server error. Earlier releases of Unified CM returned the 200 code for indicating internal server errors.
- List API added for every Handler. Specific List APIs like listPhoneByName, listPhoneByDescription, and so on are removed. A listPhone API with a search criteria that can be specified by the user is introduced.

New Information for Unified CM 7.1(2)

Unified CM (Unified CM) 7.1(2) APIs are compatible with all previous releases of Unified CM. For additional details about the Unified CM database schema changes, refer to the *Unified CM Data Dictionary for Release 7.1(2)*.

The following sections describe API updates that were made in Unified CM 7.1(2):

- New APIs, page 2-113
- Changed Operations, page 2-109

New APIs

Table 2-5 describes new operations in Unified CM 7.1(2).

Table 2-5 New Operations in Unified CM 7.1(2)

Operation	Purpose	Added Tags
addGeoLocation updateGeoLocation getGeoLocation removeGeoLocation	Added for logical partitioning feature.	name(mandatory) country description nationalSubdivision district communityName cityDivision neighbourhood street leadingStreetDirection trailingStreetSuffix streetSuffix houseNumber houseNumberSuffix landmark location floor occupantName postalCode

Table 2-5 New Operations in Unified CM 7.1(2) (continued)

Operation	Purpose	Added Tags
addGeoLocationPolicy updateGeoLocationPolicy getGeoLocationPolicy removeGeoLocationPolicy	Added for logical partitioning feature.	name(mandatory) country description nationalSubdivision district communityName cityDivision neighbourhood street leadingStreetDirection trailingStreetSuffix streetSuffix houseNumber houseNumberSuffix landmark location floor occupantName postalCode relatedPolicies, relatedPolicy, geoLocationPolicyA geoLocationPolicyAName geoLocationDeviceA geoLocationPolicyB geoLocationPolicyBName geoLocationDeviceB logicalPartitionPolicy
addGeoLocationFilter updateGeoLocationFilter getGeoLocationFilter removeGeoLocationFilter	Added for logical partitioning feature.	name description useCountry useNationalSubDivision useDistrict useCommunityName useCityDivision useNeighbourhood useStreet useLeadingStreetDirection useTrailingStreetSuffix useStreetSuffix useHouseNumber useHouseNumberSuffix useLandmark useLocation useFloor useOccupantName usePostalCode

Table 2-5 New Operations in Unified CM 7.1(2) (continued)

Operation	Purpose	Added Tags
addCommonPhoneConfig updateCommonPhoneConfig getCommonPhoneConfig removeCommonPhoneConfig	Added for phone support feature.	name description unlockPwd dndOption dndAlertingType backgroundImage phonePersonalization phoneServiceDisplay sshUserId sshPwd alwaysUsePrimeLine alwaysUsePrimeLineforVoiceMessage vendorConfig

Changed Operations

Table 2-6 describes changed operations in Unified CM 7.1(2).

Table 2-6 Changed Operations in Unified CM 7.1(2)

Operation	Change	Tags
addLine updateLine getLine	Added optional tags for park monitoring feature.	ParkMonForwardNoRetrieveDN ParkMonForwardNoRetrieveIntDN ParkMonForwardNoRetrieveIntVMEnabled ParkMonForwardNoRetrieveVMEnabled ParkMonForwardNoRetrieveCSS ParkMonForwardNoRetrieveIntCSS ParkMonReversionTimer
addLine updateLine getLine	Added optional tag for barge enchancement feature.	partyEntranceTone
addHuntPilot updateHuntPilot getHuntPilot	Added optional tags for park monitoring feature.	ParkMonForwardNoRetrieveDN ParkMonForwardNoRetrieveCSS
addH323Gateway updateH323Gateway getH323Gateway	Added optional tags for H.235 - H.323 security: voice encryption profile with native H.235/H.245 key management feature.	AllowH235PassThrough
	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tags for QSIG variant Per Trunk or Gateway feature.	ASN1ROSEOIDEncoding QSIGVariant tunneledProtocol

Table 2-6 Changed Operations in Unified CM 7.1(2) (continued)

Operation	Change	Tags
	Added optional tags for more enhancements to CPN transformations.	nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits cgpnTransformationNationalCSS cgpnTransformationInternationalCSS cgpnTransformationUnknownCSS cgpnTransformationSubscriberCSS useDevicePoolCgpnTransformNationalCSS useDevicePoolCgpnTransformInternationalCSS useDevicePoolCgpnTransformUnknownCSS useDevicePoolCgpnTransformSubscriberCSS
addH323Trunk updateH323Trunk getH323Trunk	Added optional tag for H.235 - H.323 security: voice encryption profile with native H.235/H.245 key management feature.	AllowH235PassThrough
	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tags for QSIG variant per trunk or gateway feature.	ASN1ROSEOIDEncoding QSIGVariant
	Added optional tags for more enhancements to CPN transformations.	nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits cgpnTransformationNationalCSS cgpnTransformationInternationalCSS cgpnTransformationUnknownCSS cgpnTransformationSubscriberCSS useDevicePoolCgpnTransformNationalCSS useDevicePoolCgpnTransformInternationalCSS useDevicePoolCgpnTransformUnknownCSS useDevicePoolCgpnTransformSubscriberCSS
addCommonDeviceConfig updateCommonDeviceConfig getCommonDeviceConfig	Added optional tags for IPv6 feature.	IPAddressingMode IPAddressingModePreferenceControl allowAutoConfigurationForPhones
addSIPProfile	Added optional tag for IPv6 feature.	enableAnatForEarlyOfferCalls
updateSIPProfile getSIPProfile	Added optional tag called to enhance Clear Channel (G.clear) support feature.	gClear
addProcessNode updateProcessNode getProcessNode	Added optional tag for IPv6 feature.	IPv6Name

Table 2-6 Changed Operations in Unified CM 7.1(2) (continued)

Operation	Change	Tags
addSIPRoutePattern updateSIPRoutePattern getSIPRoutePattern	Added optional tag for IPv6 feature.	dnOrPatternIPv6
addPhone updatePhone getPhone	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tags for always-use-primeline feature.	alwaysUsePrimeLine alwaysUsePrimeLineforVoiceMessage
addH323Phone updateH323Phone getH323Phone	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
addGatewayEndpoint updateGatewayEndpoint getGatewayEndpoint	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tags for QSIG variant per trunk or gateway feature.	ASN1ROSEOIDEncoding QSIGVariant
	Added optional tags to enhance the CPN transformations feature.	nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits cgpnTransformationNationalCSS cgpnTransformationInternationalCSS cgpnTransformationUnknownCSS cgpnTransformationSubscriberCSS useDevicePoolCgpnTransformNationalCSS useDevicePoolCgpnTransformInternationalCSS useDevicePoolCgpnTransformUnknownCSS useDevicePoolCgpnTransformSubscriberCSS
addMGCP updateMGCP getMGCP	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation

Table 2-6 Changed Operations in Unified CM 7.1(2) (continued)

Operation	Change	Tags
addMGCPEndpoint updateMGCPEndpoint getMGCPEndpoint	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tags for QSIG variant per trunk or gateway feature.	ASN1ROSEOIDEncoding QSIGVariant
	Added optional tags for enhancements to CPN transformations.	nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits cgpnTransformationNationalCSS cgpnTransformationInternationalCSS cgpnTransformationUnknownCSS cgpnTransformationSubscriberCSS useDevicePoolCgpnTransformNationalCSS useDevicePoolCgpnTransformInternationalCSS useDevicePoolCgpnTransformUnknownCSS useDevicePoolCgpnTransformSubscriberCSS
addSIPTrunk updateSIPTrunk getSIPTrunk	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
	Added optional tag for IPv6 feature.	destinationAddressIPv6
	Added optional tags for enhancements to CPN transformations.	unknownStripDigits cgpnTransformationUnknownCSS useDevicePoolCgpnTransformUnknownCSS
addVG224 updateVG224 getVG224	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName sendGeoLocation
addDevicePool updateDevicePool getDevicePool	Added optional tags for logical partitioning feature.	geoLocationName geoLocationFilterName
	Added optional tags for enhancements to CPN transformations.	nationalStripDigits internationalStripDigits unknownStripDigits subscriberStripDigits cgpnTransformationNationalCSS cgpnTransformationInternationalCSS cgpnTransformationUnknownCSS cgpnTransformationSubscriberCSS

Table 2-6 Changed Operations in Unified CM 7.1(2) (continued)

Operation	Change	Tags
addCommonPhoneConfig updateCommonPhoneConfig getCommonPhoneConfig	Added optional tags for always-use-primeline feature.	alwaysUsePrimeLine alwaysUsePrimeLineforVoiceMessage
addDeviceProfile updateDeviceProfile getDeviceProfile	Added optional tags for always-use-primeline feature.	alwaysUsePrimeLine alwaysUsePrimeLineforVoiceMessage

New Information for Unified CM 7.0(1)

Unified CM 7.0(1) APIs are compatible with all previous releases of Unified CM. For additional details about the Unified CM database schema changes, refer to the Unified CM *Data Dictionary for Release 7.0(1)*.

The following sections describe API updates that were made in Unified CM 7.0(1):

- New APIs, page 2-113
- Changed Operations, page 2-119

New APIs

Table 2-7 describes new operations in Unified CM 7.0(1).

Table 2-7 New Operations in Unified CM 7.0(1)

Operation	Purpose	Added Tags
addCalledPartyTransformationPattern removeCalledPartyTransformationPattern updateCalledPartyTransformationPattern getCalledPartyTransformationPattern	Added for Local Route Group feature	pattern (mandatory) usage (mandatory) routePartition description numberingPlan routeFilter patternUrgency (read only) discardDigits calledPartyTransformationMask prefixDigitsOut calledPartyNumberType calledPartyNumberingPlan

Table 2-7 New Operations in Unified CM 7.0(1) (continued)

Operation	Purpose	Added Tags
addSIPTrunkSecurityProfile updateSIPTrunkSecurityProfile removeSIPTrunkSecurityProfile getSIPTrunkSecurityProfile	Added for SRTP support for SIP Trunk feature	name (mandatory) description securityMode incomingTransport outgoingTransport digestAuthentication noncePolicyTime x509SubjectName incomingPort applLevelAuthentication acceptPresenceSubscription acceptOutOfDialogRefer allowReplaceHeader acceptUnsolicitedNotification transmitSecurityStatus
addResourcePriorityNamespace updateResourcePriorityNamespace removeResourcePriorityNamespace getResourcePriorityNamespace	Added for AS-SIP feature	namespace
addResourcePriorityNamespaceList updateResourcePriorityNamespaceList getResourcePriorityNamespaceList removeResourcePriorityNamespaceList	Added for AS-SIP feature	name (mandatory) members (mandatory) resourcePriorityNamespace / resourcePriorityNamespaceName
addResourcePriorityDefaultNamespace updateResourcePriorityDefaultNamespace getResourcePriorityDefaultNamespace removeResourcePriorityDefaultNamespace	Added for AS-SIP feature	resourcePriorityNamespace
addSIPProfile updateSIPProfile getSIPProfile removeSIPProfile	Added for AS-SIP feature	resourcePriorityNamespaceList
addTODAccess updateTODAccess getTODAccess removeTODAccess	Added for Mobility - TOD Access List feature	name (mandatory) description ownerId (mandatory) members > member > timeSchedule (mandatory) isActionAllowed (mandatory), accessList (mandatory) associatedRemoteDestination (read only)

Table 2-7 New Operations in Unified CM 7.0(1) (continued)

Operation	Purpose	Added Tags
getMobileSmartClientProfile	Added for Unified CM new device type	MobileSmartClient Name EnableSNRUri EnableCFAUri HandoffUri
addVG224 updateVG224 removeVG224 getVG224	Added for adding VG224 gateway	domainName description product protocol model callManagerGroup callManagerGroupName units unit product

Table 2-7 New Operations in Unified CM 7.0(1) (continued)

Operation	Purpose	Added Tags
		subunits subunit endpoints endpoint name description product productInfo model modelInfo class protocol protocolSide callingSearchSpace callingSearchSpaceName devicePool devicePoolName commonDeviceConfig commonDeviceConfigName commonPhoneConfig commonPhoneConfigName networkLocation location, locationName mediaResourceList mediaResourceListName networkHoldMOHAudioSourceId userHoldMOHAudioSourceId automatedAlternateRoutingCSS automatedAlternateRoutingCSSName aarNeighborhood aarNeighborhoodName loadInformation vendorConfig versionStamp traceFlag mlppDomainId mlppIndicationStatus preemption useTrustedRelayPoint retryVideoCallAsAudio securityProfile securityProfileName sipProfile sipProfileName cgpnTransformationCSS cgpnTransformationCSSName

Table 2-7 New Operations in Unified CM 7.0(1) (continued)

Operation	Purpose	Added Tags
		useDevicePoolCgpnTransformCSS lines packetCaptureMode packetCaptureDuration transmitUTF8 ports userLocale networkLocale isActive unattendedPort subscribeCallingSearchSpace subscribeCallingSearchSpaceName allowCtiControlFlag remoteDevice phoneTemplate phoneTemplateName presenceGroup presenceGroupName ignorePresentationIndicators deviceMobilityMode hlogStatus ownerUserId vendorConfig versionStamp
addApplicationUser	Added for creating new application users	userid password passwordCredentials digestCredentials presenceGroup presenceGroupName acceptPresenceSubscription acceptOutOfDialogRefer acceptUnsolicitedNotification allowReplaceHeader isStandard associatedDevices associatedGroups associatedCAPFProfiles

Table 2-7 New Operations in Unified CM 7.0(1) (continued)

Operation	Purpose	Added Tags
updateApplicationUser	Added for updating application users	userid password passwordCredentials digestCredentials presenceGroup presenceGroupName acceptPresenceSubscription acceptOutOfDialogRefer acceptUnsolicitedNotification allowReplaceHeader associatedDevices associatedGroups
removeApplicationUser	Added for removing existing application users	userid
getApplicationUser	Added for obtaining details about application users	userid

Changed Operations

Table 2-8 describes changed operations in Unified CM 7.0(1).

Table 2-8 Changed Operations in Unified CM 7.0(1)

Operation	Change	Tags
addDevicePool updateDevicePool getDevicePool	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS nationalPrefix internationalPrefix unknownPrefix subscriberPrefix
	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
addMGCPEndPoint	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS
	Added tag for Network Virtualization feature	useTrustedRelayPoint
	Added tag for Secure-Indication Tone feature	enableProtectedFacilityIE

Table 2-8 Changed Operations in Unified CM 7.0(1) (continued)

Operation	Change	Tags
addSIPTrunk updateSIPTrunk getSIPTrunk	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS unknownPrefix
	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
	Added tags for Network Virtualization feature	useTrustedRelayPoint
	Added tags for SRTP support feature	srtpAllowed srtpFallbackAllowed
	Added tags for SIP PAI feature	isPaiEnabled sipPrivacy isRpidEnabled sipAssertedType
	Added tag for Trunk Licensing feature	licensedCapacity
addH323Phone updateH323Phone getH323Phone	Added tag for Network Virtualization feature	useTrustedRelayPoint
addH323Trunk updateH23Trunk getH323Trunk	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS nationalPrefix internationalPrefix unknownPrefix subscriberPrefix
	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
	Added tags for Network Virtualization feature	useTrustedRelayPoint
	Added tags for Trunk Licensing feature	licensedCapacity
addH323Gateway updateH323Gateway getH323Gateway	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS nationalPrefix internationalPrefix unknownPrefix subscriberPrefix
	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
	Added tag for Network Virtualization feature	useTrustedRelayPoint
	Added tag for Trunk Licensing feature	licensedCapacity

Table 2-8 Changed Operations in Unified CM 7.0(1) (continued)

Operation	Change	Tags
addRoutePattern updateRoutePattern getRoutePattern	Added tags for CPN and E.164 Dialing feature	callingPartyNumberingPlan allingPartyNumberType
	Added tags for Local Route Group feature	calledPartyNumberingPlan calledPartyNumberType
	Added tag for AS-SIP feature	resourcePriorityNamespace
addHuntPilot updateHuntPilot getHuntPilot	Added tags for CPN and E.164 Dialing feature.	callingPartyNumberingPlan callingPartyNumberType
	Added tags for Local Route Group feature	calledPartyNumberingPlan calledPartyNumberTyp
addTransPattern updateTransPattern getTransPattern	Added tags for CPN and E.164 Dialing feature	callingPartyNumberingPlan callingPartyNumberType
	Added tags for Local Route Group feature	calledPartyNumberingPlan calledPartyNumberType
	Added tag for AS-SIP feature	resourcePriorityNamespace
addConferenceBridge updateConferenceBridge getConferenceBridge	Added tag for Network Virtualization feature	useTrustedRelayPoint
addCTIRoutePoint updateCTIRoutePoint getCTIRoutePoint	Added tagsAdded tag for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS
	Added tag for Network Virtualization feature	useTrustedRelayPoint
addPhone updatePhone getPhone	Added tags for CPN and E.164 Dialing feature	cgpnTransformationCSS useDevicePoolCgpnTransformCSS
	Added tags for BLF CallPickup feature	associatedBLFSDFeatures ringSettingIdleBLFAudibleAlert ringSettingBusyBLFAudibleAlert
	Added tags for Enhanced IP Phone Services Provisioning feature	phoneService vendor version priority phoneServiceDisplay requirePKIAuthForHTTPS
	Added tag for Secure-Indication Tone feature	isProtected
	Added tag for new CUCM Device type feature	MobileSmartClientProfile
	Added tags for complete phone API support	requireDTMFReception RFC2833Disabled phoneLoadName authenticationMode keySize

Table 2-8 Changed Operations in Unified CM 7.0(1) (continued)

Operation	Change	Tags
addGatewayEndpoint updateGatewayEndpoint getGatewayEndpoint	Added tags for CPN and E.164 Dialing feature	gpnTransformationCSS useDevicePoolCgpnTransformCSS nationalPrefix internationalPrefix unknownPrefix subscriberPrefix
	Added tags for Local Route Group feature	cdpnTransformationCSS useDevicePoolCdpnTransformCSS
	Added tag for Network Virtualization feature	useTrustedRelayPoint
	Added tag for VoSIP/DVX G.Clear feature	GClearEnable
addMOHServer updateMOHServer getMOHServer	Added tag for Network Virtualization feature	useTrustedRelayPoint
addVoiceMailPort getVoiceMailPort		
addCommonDeviceConfig updateCommonDeviceConfig getCommonDeviceConfig		
addTranscoder updateTranscoder getTranscoder	Added tag for Network Virtualization feature	isTrustedRelayPoint
addRemoteDestinationProfile updateRemoteDestinationProfile getRemoteDestinationProfile	Added tags for DND Reject feature	dndStatus dndOption
	Added tag for new CUCM Device type feature	MobileSmartClientProfile
addTransformationPattern updateTransformationPattern getTransformationPattern	Added tags for CPN and E.164 Dialing feature	callingPartyNumberingPlan digitDiscardInstruction callingPartyNumberTyp
addTimePeriod updateTimePeriod getTimePeriod	Added tags for Mobility - TOD Access List feature	description isPublished todOwnerId dayOfMonthEnd monthOfYearEnd
addTimeSchedule updateTimeSchedule getTimeSchedule	Added tags for Mobility - TOD Access List feature	description Published timeScheduleCategory todOwnerId

Table 2-8 Changed Operations in Unified CM 7.0(1) (continued)

Operation	Change	Tags
addRemoteDestination	Added tags for Mobility - TOD Access List feature	timeZone clientApplicationModel todAccess
	Removed tags for Mobility - TOD Access List feature	allowedAccessList blockedAccessList smartClientInstalled
	Added tag for new CUCM Device type feature	MobileSmartClient
	Deprecated tag	clientAppModelxxx
updateRemoteDestination getRemoteDestination	Added tags for Mobility - TOD Access List feature	timeZone clientApplicationModel todAccess
	Removed tags for Mobility - TOD Access List feature	allowedAccessList blockedAccessList smartClientInstalled
	Added tag for new CUCM Device type feature	MobileSmartClient
	Added tag for new CUCM Device type feature	clientAppModelxxx
addUser updateUser getUser	Added tag for Mobility - TOD Access List feature	associatedTodAccess
	Removed tag for addUser API for Mobility - TOD Access List feature	associatedAccessLists
addRouteList updateRouteList getRouteList	Added tags for CPN and E.164 Dialing feature	callingPartyNumberingPlan callingPartyNumberType calledPartyNumberingPlan calledPartyNumberType

Dynamic Throttling of Requests

Unified CM 7.0 includes a new throttling mechanism called dynamic throttling of request. The MaxAXLWritesPerMinute service parameter, used in earlier releases, has been deprecated. For more information see, Dynamic Throttling of Requests, page 2-138.

New Information for Unified CM 6.1 (1)

Table 2-9 describes the API calls that changed in Unified CM 6.1(1). These changes may require updates to the existing user-code that uses these APIs.

Table 2-9 Changed API Calls in Unified CM 6.1(1)

API Call	Change	Tags
addLine	Added tag for Intercom CTI Support feature	defaultActivatedDevice
updateLine	Added tag for Intercom CTI Support feature	defaultActivatedDevice

Table 2-9 Changed API Calls in Unified CM 6.1(1) (continued)

API Call	Change	Tags
getLine	Added tag for Intercom CTI Support feature	defaultActivatedDevice
addUser	Added tag for Mobility user feature	primaryDevice
updateUser	Addedtag for Mobility user feature	primaryDevice
getUser	Added tag for Mobility user feature	primaryDevice
addDeviceProfile	Added tags for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
updateDeviceProfile	Added tags SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
getDeviceProfile	Added tags for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
addDevicePool	Added tags for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
updateDevicePool	Added tags for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
getDevicePool	Added tags for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
addPhone	Added tags and joinAcrossLines for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
	Added tag for BAT/TAPS Licensing Allowance feature	isActive
updatePhone	Added tags and joinAcrossLines for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
	Added tag for BAT/TAPS Licensing Allowance feature	isActive
getPhone	Added tags and joinAcrossLines for SingleButtonBarge and JoinAcrossLines features	singleButtonBarge joinAcrossLines
	Added tag for BAT/TAPS Licensing Allowance feature	isActive

Note All Unified CM 6.0 AXL methods, with the exception of ExecuteSQLQuery and ExecuteSQLupdate, are backward compatible with Unified CM 6.1. By default, the interface automatically uses the 6.0 AXL schema. Developers should specify SOAPAction: “CUCM:DB ver=6.1” in the HTTP header to use any new 6.1 methods.

New Information for Unified CM 6.0(1)

For Unified CM Release 6.0(1), be aware that the defined AXL APIs are backward compatible. However, the executeSQLQuery request, which lets the user run a database query directly on the Unified CM database, is not backward compatible.

- Release 6.0(1) splits some of the tables in the Unified CM database. Feature-related information moved to the corresponding dynamic tables. If the direct SQL query to which the 'executeSQLQuery' API referred uses any of the changed tables, you may need to rewrite the query according to the new database schema.
- The columns enduser.password and enduser.pin from the enduser table and the applicationuser.password column from the applicationUser table moved to the credential table as credential.credentials. A direct SQL query that refers to these columns will not work in Unified CM Release 6.0(1).

Note Be aware that Unified CM password and pin fields are encrypted. Applications should not write to those fields using <executeSQLUpdate>. Instead, update passwords and pins by using the appropriate <updateXXXUser> request.

- The phone API for extension mobility-related parameters has the new tag CurrentConfig. This tag is valid only for the getPhone response. The tag lets AXL provide the original device configuration and the logged-in profile information:
  - If a user has logged in to a device by using a device profile, the CurrentConfig tag contains the values for the extension mobility-related parameters from that device profile.
  - If no user has logged in, the CurrentConfig tag contains the values of the extension mobility-related parameters for the actual device.
- Schema changes for the CMCInfo and FACInfo APIs help maintain consistency with other AXL APIs.

For further details about the Unified CM database schema changes, refer to Unified CM *Data Dictionary for Release 6.0(1)*.

Note The getCCMVersion API will return the Unified CM version based on the Node Name that is specified in the request. If no Node Name is specified, you will get the Unified CM Version of the lowest node ID Unified CM.

Cisco always advises running the AXL API on a completely upgraded cluster. When run on a cluster that is not upgraded completely, the response of the AXL API will be correct when executed on a server that already has been upgraded. However, if you execute the AXL API on a server that has not yet been upgraded, then it will return the Unified CM Version of the lowest node ID Unified CM per the server local database information.

The following AXL API calls changed in Unified CM Release 6.0(1). These changes may require changes to the existing user-code that uses these APIs:

- updateAppUser
- addCallPickupGroup
- updateCallPickupGroup
- getCallPickupGroup
- addConferenceBridge

- updateConferenceBridge
- getConferenceBridge
- addCSS
- updateCSS
- getCSS
- addDevicePool
  - In axl.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in XDevicePool changed to match the AXL response.
  - In axlsoap.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in updateDevicePoolReq changed.
- updateDevicePool
  - In axl.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in XDevicePool changed to match the AXL response.
  - In axlsoap.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in updateDevicePoolReq changed.
- getDevicePool
  - In axl.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in XDevicePool changed to match the AXL response.
  - In axlsoap.xsd, the tag name aarNeighborhood and the annotation for the revertPriority tag in updateDevicePoolReq changed.
- addDeviceProfile
- updateDeviceProfile
- getDeviceProfile
- addGatewayEndpoint
- updateGatewayEndpoint
- getGatewayEndpoint
- addH323Phone
- updateH323Phone
- getH323Phone
- addH323Trunk
- updateH323Trunk
- getH323Trunk
- addLine
- updateLine
- getLine
- addMGCP
- getMGCP
- addPhone
- updatePhone
- getPhone
- addRegion

- updateRegion
- getRegion
- updateRegionMatrix
- addRoutePartition
- updateRoutePartition
- getRoutePartition
- addSIPTrunk
- updateSIPTrunk
- getSIPTrunk
- addUser
- updateUser
- getUser
- addVoiceMailPort
- updateVoiceMailPort
- getVoiceMailPort
- doAuthenticateUser
- getCMCInfo, removeCMCInfo, and updateCMCInfo

  In axlsoap.xsd

  – A new option tag “code” along with the “uuid” tag for these three requests exist. The user can send either the uuid or code tag.

  – This release renames the existing tag “code” to “newCode” in the updateCMCInfo request. Users can send the new code to be updated as the “newCode” tag instead of “code” in update CMCInfo requests.

  – This release removes the invalid authorizationLevel tag in the updateCMCInfo request.

- addFACInfo, getFACInfo, updateFACInfo, and removeFACInfo

  In axl.xsd

  – The existing tag “description” changed to “name.” This makes the addFACInfo, getFACInfo, and updateFACInfo request match the database. In previous releases, the value that was supplied for the “description” tag updated “name” in the database.

  In axlsoap.xsd

  – A new option tag “name” along with the “uuid” tag for getFACInfo, updateFACInfo, and removeFACInfo exist.

  – The existing tag “description” changed to “newName” in the updateFACInfo request.

  – Users can send either uuid or name in the getFACInfo, updateFACInfo, and removeFACInfo requests.

This release deprecated some of the fields that were removed from the Unified CM 6.0(1) database in AXL. This release adds annotation for such fields.

AXL Versioning Support

To improve backward compatibility, Unified CM introduced AXL schema versioning in Release 6.0(1). This feature is included in all subsequent Unified CM releases. Beginning with Release 6.0(1), the system duplicates the previous AXL 1.0 schema as the AXL 6.0(1) schema and numbers the AXL schema the same as the corresponding Unified CM release. This approach maintains AXL backward compatibility for one full release cycle.

Changed Service Parameter for Unified CM 6.0(1)

Unified CM 6.0(1) adds a new service parameter, EnableAXLEncodingInfo, to the Unified CM Administrator windows under the Cisco Database Layer Monitor service. This parameter allows the user to decide whether AXL responses should contain the encoding information. Consider encoding information as important if an AXL request has non-English characters in it.

Unified CM 5.1(1) added a new service parameter, Send Valid Namespace in the AXL response, under the Cisco Database Layer Monitor service. This parameter determines the namespace that is sent in the AXL response from the Unified CM. In the 6.0(1) release, the default value of this parameter changed from False to True.

- When this parameter is True, Unified CM sends the valid namespace in the AXL response, so the namespace matches the AXL schema specification.
- If the parameter is False, Unified CM sends an invalid namespace in the AXL response, which does not match the AXL schema specification.

  To maintain backward compatibility with older applications, you might need to change the value to False. Cisco recommends that you set this parameter to True, so the Unified CM sends a valid namespace.

New Information for Unified CM 5.1(1)

The following list provides AXL API calls that are new in Unified CM 5.1(1):

- addSIPRealm
- updateSIPRealm
- getSIPRealm
- removeSIPRealm

These APIs add and update credentials (passwordreserve) in siprealm.

In addition, Unified CM Administration 5.1 release adds a new service parameter, “Send Valid Namespace in AXL Response,” under the Cisco Database Layer Monitor service. This parameter determines the namespace that gets sent in the AXL response from Unified CM.

When this parameter specifies True, Unified CM sends the valid namespace in the AXL response so the namespace matches the AXL schema specification.

If the parameter specifies False, Unified CM sends an invalid namespace in the AXL response, which does not match the AXL schema specification.

The default service parameter value specifies False to maintain backward compatibility with the AXL response in the Unified CM 5.0 release. Cisco recommends that you set this parameter to True so Unified CM sends the valid namespace.

New Information for Unified CM 5.0(1)

The following AXL API calls are new in Unified CM 5.0(1):

- executeSQLUpdate
- doAuthenticateUser
- updateAppUser
- addUserGroup
- updateUserGroup
- removeUserGroup
- getUserGroup

The following AXL API calls have been changed in Unified CM 5.0:

- addPhone
- updatePhone
- getPhone
- addGatewayEndpoint
- updateGatewayEndpoint
- getGatewayEndpoint
- addMGCPEndpoint
- updateMGCP
- addSIPTrunk
- updateSIPTrunk
- getSIPTrunk
- addCallManager
- updateCallManager
- getCallManager
- addCallPark
- addRoutePattern
- updateRoutePattern
- updateTransPattern
- updateHuntPilot
- addHuntList
- updateHuntList
- getHuntList
- addPilotPoint
- updatePilotPoint
- getPilotPoint
- addH323Gateway
- updateH323Gateway

- updateH323Phone
- getH323Gateway
- getH323Trunk
- addUser
- updateUser
- getUser
- updateProcessNodeService
- getProcessNodeService
- doDeviceLogout
- listUserByName
- updateServiceParameter
- updateGatekeeper
- updateConferenceBridge
- updateAttendantConsoleHuntGroup
- updateDeviceProfile
- updateLine
- updateLineGroup
- addDevicePool
- updateDevicePool
- doDeviceReset

The following API calls that have been deprecated in Unified CM 5.0:

- addDDI
- updateDDI
- removeDDI
- addDialPlan
- updateDialPlan
- removeDialPlan
- addDialPlanTag
- updateDialPlanTag
- removeDialPlanTag

New Information for Unified CM 4.2(2)

This section describes the new or changed API calls for Unified CM 4.2(2) and a new service parameter for Cisco Database Layer Monitor.

Changed API Calls

Changes to the following API calls in Unified CM 4.2(2) support the Hold Reversion feature:

- addDevicePool (adds revertPriority to this request)

- updateDevicePool (adds revertPriority to this request)
- addLine (adds hrDuration and hrInterval to this request)
- updateLine (adds hrDuration and hrInterval to this request)

Because these new tags are disabled by default, these changes are backward-compatible with existing user code.

New Service Parameter

A new service parameter, EnableAXLEncodingInfo, has been added to Unified CM Administration under the Cisco Database Layer Monitor service. This parameter enables the user to decide if AXL responses should contain encoding information. Encoding information is important if an AXL request has non-English characters in it.

New Information for Unified CM 4.1(2)

The following AXL API calls are new in Unified CM 4.1(2):

addTimePeriod

updateTimePeriod

removeTimePeriod

getTimePeriod

addTimeSchedule

updateTimeSchedule

removeTimeSchedule

getTimeSchedule

addCMCInfo

updateCMCInfo

removeCMCInfo

getCMCInfo

addFACInfo

updateFACInfo

removeFACInfo

getFACInfo

The following AXL API calls have been changed in Unified CM 4.1(2):

addPhone

updatePhone

getPhone

addLine

updateLine

addUser

removeUser

updateUser

getUser
addDeviceProfile
updateDeviceProfile
getDeviceProfile
addRoutePattern
updateRoutePattern
getRoutePattern
addRouteList
updateRouteList
getRouteList
addGatewayEndpoint
updateGatewayEndpoint
getGatewayEndpoint
addH323Trunk
updateH323Trunk
removeH323Trunk
getH323Trunk
addHuntPilot
updateHuntPilot
removeHuntPilot
getHuntPilot
addProcessNode
updateProcessNode
removeProcessNode
getProcessNode
listAllProcessNodes
listProcessNodesByService
addRoutePartition
updateRoutePartition
removeRoutePartition
getRoutePartition
addH323Gateway
updateH323Gateway
removeH323Gateway
getH323Gateway
addH323Phone
updateH323Phone
removeH323Phone

getH323Phone

addHuntList

updateHuntList

removeHuntList

getHuntList

AXL Schema Documentation

The axlsqltoolkit.zip plug-in contains the following five AXL schema files:

- AXLAPI.wsdl
- AXLEnums.xsd
- axlmessage.xsd
- axlsoap.xsd
- axl.xsd

These files encapsulate the complete AXL schema, including details of all requests, responses, XML objects, and data types.

In addition to these schema files, two folders exist:

- WSDL-AXIS
- WSDL-NET

Each of these folders contains AXLAPI.wsdl and AXLSoap.xsd files to be used for application development in AXIS or .NET client environments, respectively.

You can obtain complete documentation of all available AXL messages from the Cisco Developer Services web site: http://developer.cisco.com. This website requires a Cisco.com login.

You can use the ***Application > Plugins > Unified CM AXL SQL Toolkit*** command from the Unified CM server administration interface to obtain:

- AXL schema (.xsd) files

  See also AXL Schema Documentation, page 2-133.

- The WSDL file

AXL Versioning Support

To improve backward compatibility, Unified CM introduced AXL schema versioning in Release 6.0(1). This feature is included in all subsequent Unified CM releases. Beginning with Release 6.0(1), the system duplicates the previous AXL 1.0 schema as the AXL 6.0(1) schema and numbers the AXL schema the same as the corresponding Unified CM release. This approach maintains AXL backward compatibility for one full release cycle.

Cisco highly recommends that developers include the version of AXL schema on which an AXL request is based because support for unversioned requests might be removed in future releases of Unified CM.

For those developers who are using the AXL APIs executeSQLQuery and executeSQLUpdate, changes have occurred to the Unified CM database schema that affect the direct SQL query approach. Refer to the *Unified CM Database Dictionary,* at http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_reference_guides_list.html, for the release that you are using. That document describes the specific changes in the database schema.

To help developers plan for AXL versioning, Table 2-10 provides the approach that Cisco will follow in supporting upcoming releases.

- Cisco will support AXL requests ***without*** version information for only three releases following the 6.0(1) release; after that, requests without version information will be rejected.
- AXL requests ***with*** version information will have the corresponding schema applied for up to three subsequent releases; after that, the specified version may not be available.

AXL Policy Considerations

The following policies related to AXL versioning support is under consideration for future release:

- AXL schema versioning will continue indefinitely.
- AXL schemas will be available for two major Unified CM release cycles, such that AXL applications will require minor updates every two years.
- Developers who do not request a specific AXL schema will always connect to the oldest schema available.
- Developers can request other available AXL schemas by specifying the AXL schema version in the SOAP Action Header.

Table 2-10 AXL Versioning and Schema Plan

		AXL Request no version specified	AXL Request with Version Specified								
			CUCM:DB ver=6.0(	CUCM:DB ver=6.1	CUCM:DB ver=7.0	CUCM:DB ver=7.1	CUCM:DB ver-8.0	CUCM:DB ver-8.5	CUCM:DB ver-9.0	Plus 1 release	Plus 2 releases
Cisco Unified CM Release	**Release 6.0(1)**	6.0 schema applied	6.0 schema applied	Not Applicable							
	Release 6.1(0)	6.0 schema applied	6.0 schema applied	6.1 schema applied							
	Release 7.0(1)	6.0 schema applied	6.0 schema applied	6.1 schema applied	7.0 schema applied						
	Release 7.1(2)	6.0 schema applied	6.0 schema applied	6.1 schema applied	7.0 schema applied	7.1 schema applied					
	Release 8.0(1)	6.1 schema applied	Schema no longer available	6.1 schema applied	7.0 schema applied	7.1 schema applied	8.0 schema applied				
	Release 8.5(1)	7.0 schema applied			7.0 schema applied	7.1 schema applied	8.0 schema applied	8.5 schema applied			
	Release 9.0(1)	7.1 schema applied				7.1 schema applied	8.0 schema applied	8.5 schema applied	9.0 schema applied		
	Plus 1 release	8.0 schema applied					8.0 schema applied	8.5 schema applied	9.0 schema applied	Plus 1 release schema applied	
	Plus 2 releases	8.5 schema applied						8.5 schema applied	9.0 schema applied	Plus 1 release schema applied	Plus 2 releases schema applied

Note Release 8.6(x) does not include a new schema. Developers should use the 8.5 schema with Release 8.6(x). All new 8.6 database objects will be added to the 9.0 schema in Release 9.0(1)

The following sample AXL request carries version information:

```
Host:10.77.31.194:8443
Authorization: Basic Q0NNQWRtaW5pc3RyYXRvcjpjaXNjb19jaXNjbw==
Accept: text/*
Content-type: text/xml
SOAPAction: "CUCM:DB ver=7.0"
Content-length: 427
SOAP-ENV:Envelope xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <axl:getUser xmlns:axl="http://www.cisco.com/AXL/7.0"
si:schemaLocation="http://www.cisco.com/AXL/API/7.0 http://ccmserver/schema/axlsoap.xsd"
sequence="1234">
      <userid>tttt</userid>
    </axl:getUser>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample AXL response:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONIDSSO=12E52A7F9B34107BA6147096878E00F9; Path=/
Set-Cookie: JSESSIONID=7B94F17D61C6A91B04C5C76A6E3F905E; Path=/axl; Secure
SOAPAction: "CUCM:DB ver=7.0"
Content-Type: text/xml;charset=utf-8
Content-Length: 1936
Date: Mon, 03 Mar 2008 10:17:38 GMT
Connection: close
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="
http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Header/>

<SOAP-ENV:Body>
  <axl:getUserResponse xmlns:axl="http://www.cisco.com/AXL/API/7.0"
xmlns:xsi="http://www.cisco.com/AXL/API/7.0" sequence="1234">
    <return>
      <user>
        <firstname/>
        <lastname>tttt</lastname>
        <userid>tttt</userid>
        <password/>
        <pin/>
        <telephoneNumber/>
        <department/>
        <manager/>
        <associatedDevices>
          <device>SEPA88888888888</device>
        </associatedDevices>
        <primaryExtension/>
        <associatedPC/>
        <associatedGroups>
          <userGroup uuid="{6B126A13-8F47-B78D-13D4-9555D664F634}">
     <name>test</name>
       <userRoles>
         <userRole uuid="{A6BAE213-AAB5-F794-B71C-98EE94129C9B}">Standard AXL API
Access</userRole>
       </userRoles>
          </userGroup>
        </associatedGroups>
        <enableCTI>true</enableCTI>
        <digestCredentials/>
```

```
        <enableMobility>false</enableMobility>
        <enableMobileVoiceAccess>false</enableMobileVoiceAccess>
        <maxDeskPickupWaitTime>10000</maxDeskPickupWaitTime>
        <remoteDestinationLimit>4</remoteDestinationLimit>
        <passwordCredentials>
          <pwdCredPolicyName>Default Credential Policy</pwdCredPolicyName>
          <pwdCredUserCantChange>false</pwdCredUserCantChange>
          <pwdCredUserMustChange>false</pwdCredUserMustChange>
          <pwdCredDoesNotExpire>false</pwdCredDoesNotExpire>
          <pwdCredTimeChanged>February 14, 2008 16:10:12 IST</pwdCredTimeChanged>
          <pwdCredTimeAdminLockout/>
          <pwdCredLockedByAdministrator>false</pwdCredLockedByAdministrator>
        </passwordCredentials>
        <pinCredentials>
          <pinCredPolicyName>Default Credential Policy</pinCredPolicyName>
          <pinCredUserCantChange>false</pinCredUserCantChange>
          <pinCredUserMustChange>false</pinCredUserMustChange>
          <pinCredDoesNotExpire>false</pinCredDoesNotExpire>
          <pinCredTimeChanged>February 14, 2008 16:10:12 IST</pinCredTimeChanged>
          <pinCredTimeAdminLockout/>
          <pinCredLockedByAdministrator>false</pinCredLockedByAdministrator>
        </pinCredentials>
      </user>
    </return>
  </axl:getUserResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Authentication

The system controls user authentication via the HTTPS Basic Authentication scheme; therefore, you must include the Authorization header in the HTTPS header. Because Base64 encoding takes three 8-bit bytes and represents them as four printable ASCII characters, if the encoded header does not contain an even multiple of four ASCII characters (16, 20, 24, and so on), you must add padding characters (=) to complete the final group of four.

Ensure users are authorized to access AXL. For help with configuring authorization, see Post-Installation Steps and Troubleshooting on the Linux Platform, page 2-146.

If user authentication of the user fails, the system returns an HTTP 401 Access Denied error to the client.

For example, if the user agent wants to send the userid “larry” and password “curly and moe,” it would use the following header field:

```
Authorization: Basic bGFycnk6Y3VybHkgYW5kIG1vZQ==
```

where the string “`bGFycnk6Y3VybHkgYW5kIG1vZQ==`” provides the Base64 encoding of “larry:curly and moe.”

Note The two “equals” characters (=) at the end of the string act as padding characters for Base64 encoding.

Data Encryption

Encrypt AXL SOAP messages by using HTTP Secure Sockets Layer (SSL). SSL remains functional on the web server by default. Ensure AXL requests are made by using the “https” protocol.

Dynamic Throttling of Requests

Unified CM releases earlier than 7.0 included the AXL service parameter MaxAXLWritesPerMinute, which has a default value of 50 and maximum value of 999. This service parameter designates the maximum number of write requests that AXL can encounter and process in one minute. Unified CM 7.0 includes a new throttling mechanism. The MaxAXLWritesPerMinute service parameter has been deprecated.

The new throttling mechanism takes into account the dynamic state of Unified CM. It considers the number of outstanding change notifications across the Unified CM cluster at any given time. If a node has more than 1,500 outstanding change notifications, AXL stops processing write requests until the outstanding change notifications are below 1,500. During throttling, HTTPS Status-Code: 503 Service Unavailable response and sets AXL performance counters which can be viewed using RTMT. When a 503 Service Unavailable response is returned, Cisco recommends that the application sleep for a number of seconds or milliseconds (as determined by the developer) to allow pending write requests to be processed. The application should then continue submitting requests.

There are two AXL performance counters:

- ThrottleCount—Determines number of times Administrative AXL throttling has been engaged.
- ThrottleState—Determines the state of AXL throttling. That is, whether AXL throttling is currently active (throttling is engaged).

The AXL error message for throttling remains same as in earlier versions of Unified CM. There is no change required to AXL applications.

For example, consider an application that makes 1,000 phone insertions in 30 seconds. Assume that these insertions cause 2,000 change notifications to various applications such as Unified CM and Cisco TFTP, and that within 10 seconds all change notifications are consumed. In this situation, by the 40th second, the number of outstanding change notifications is zero and the throttling mechanism does not take effect. However, if these change notifications are not consumed, the throttling mechanism does take effect and write requests are throttled until the outstanding change notification value falls below 1,500.

The throttling mechanism considers the capacity of the Unified CM cluster to consume the change notifications that generated from all the write activities to the Unified CM database. In this way, it is dynamic. As long as all change notifications are consumed at a rate that is equal to or higher than the rate at which change notifications are generated, throttling does not take effect.

Table 2-11 AXL Query Limits

Writes Per Minute	Maximunm of 1500 Write requests
Reads Per Minute	No limit for Read requests.
Total Records	No limits for total number of records. But size of total recordes must be less than 8MB per request and 16MB is the maximum buffer allocated for parallel processing of requests.

Note

- Read requests are never throttled and pass through even when write requests are throttled.
- The MaxAXLWritesPerMinute service parameter has been deprecated and is no longer available.

AXL Data Throttling

The AXL interface now includes a throttling mechanism that limits the amount of data that client can return. The limit for data that gets returned in any single request specifies 8 MB. The limit for concurrent data requests specifies 16 MB, which can be split over any number of concurrent requests (for example, 8 concurrent requests with each requiring 2 MB of data to be returned; 4 concurrent requests with each requiring 4 MB of data to be returned; or any other combination not exceeding 8 MB per request or 16 MB concurrently).

This feature:

- Prevents AXL request processing from making the Tomcat service unresponsive.
- Allows critical applications such as the Cisco Unified Communications Manager Administration interface and logging in to and out of Cisco Extension Mobility to function even when heavy AXL queries are processed.
- Allows client applications to obtain the information that is requested.
- Maximizes interface throughput.
- Minimizes required changes to existing applications.

The following AXL methods now include data throttling:

- executeSQLQuery
- listDeviceByNameAndClass
- listDeviceByServiceName
- listPhoneByDescription
- listPhoneByName
- listUserByName

In Unified CM 8.0(1) data throttling is included only in executeSQLQuery.

With data throttling, the AXL interface now returns the following message when the 8 MB limit is reached: “Query request too large. Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>.”

In addition, new <skip> and <first> tags for the List X methods allow developers to fetch the data requested. Developers who use ExecuteSQLQuery should use standard SQL Skip and First tags in the request to retrieve the desired data set.

Interaction

This section describes how the interface responds in a variety of situations.

Example 1, Option 1: Client requests data that exceeds 8 MB

Server response: ProcessingConstraintException AXL error code 5011“Query request too large. Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>”

<Number of Rows> specifies the suggested number of rows that a single request can return to keep the data exchange under the 8-MB limit. Clients should use <Matched Rows> to determine the number of iterations that are required to retrieve the complete data set that a query tries to fetch.

Client response, Option 1:

1. Client logic analyzes the server response and obtains the value of <Row Fetch>.

2. Client stores the <Row Fetch> value in a constant that depicts Row Fetch Step Size. For this example, rowFetchStepSize represents the constant name.
3. Client logic generates the request in parts based on the <Row Fetch> value.
4. Client keeps track of the number of rows that were received in the previous request. For this example, this information exists in a variable that is named prevRows.
5. Client keeps track of the total number of rows that are fetched. For this example, this information exists in a variable that is named totalRowFetch.
6. Before sending the next request, client checks whether prevRows == <Row Fetch>.
7. If the check returns true, continue the request generation loop; otherwise, break from the loop.

Example 1, Option 2: Client requests data that exceeds 8 MB

Server response: ProcessingConstraintException AXL error code 5011 “Query request too large. Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>”

<Number of Rows> specifies the suggested number of rows that a single request can return to keep the data exchange under the 8-MB limit. Clients should use <Matched Rows> to determine the number of iterations that are required to retrieve the complete data set that a query tries to fetch.

Client response, Option 2:

1. Client logic analyzes the server response and obtains the value of <Row Fetch> and <totalRows>.
2. Client stores the <Row Fetch> and <totalRows> values in variables. For this example, the variable names comprise rowFetchStepSize and countOfRows.

Note Alternatively, the client can execute a query for count(*) to obtain the number of rows that can be fetched from the database.

3. Client logic calculates the number of iterations based on the values that are stored in rowFetchStepSize and countOfRows.
4. Client logic declares two variables: “skip” (with the initial value of 0) and “first” (with initial value set to rowFetchStepSize).
5. Client starts the iteration.
6. Client logic generates the request based on “skip” and “first” at every iteration.
7. Client modifies the value of “skip” and “first” at every iteration.
8. Client checks the response at every iteration.
9. If the response is a MemoryConstraintException, the client waits until the requests in progress completes, then continues the iteration.

   If the response is not a MemoryConstraintException, the client continues the iteration.
10. Iterations continue until the <number Of Iterations> value is reached.

Example 2: Client sends single 8-MB request

Server responds with requested data.

Example 3: Client sends two 8-MB requests simultaneously

Server responds with requested data.

Example 4: Client sends more than two 8-MB requests simultaneously

The server processes the first two requests. Other concurrent requests that may be received generate this exception: MemoryConstraintException AXL error code 5009: "Maximum AXL Memory Allocation Consumed. Please retry once requests in progress have completed."

Client response:

1. Client waits for the requests that are being processed to complete and then sends the request again.

   Cisco recommends that applications wait 2 to 3 minutes before resubmitting the request.

2. Client logic must track the requests that fail and send them again.

Example 5: Concurrent data requests reach 16-MB limit

This example applies to all AXL methods.

This situation returns MemoryConstraintException AXL error code 5009: “Maximum AXL Memory Allocation Consumed. Please retry once requests in progress have completed.”

Cisco recommends that applications wait 2 to 3 minutes after receiving this message before resubmitting the request.

Using <skip> and <first> Tags in List APIs

The new <skip> and <first> tags provide additional functionality when data is retrieved by using the List methods. If a List Response exceeds 8 MB, the client can fetch data in sets of rows by using a combination of these tags. These tags provide navigation functionality. Be aware that they are not mandatory and have no default values.

In addition,

- Negative values for the <skip> or <first> tags cause a SQL syntax error.
- If the <skip> tag is not mentioned or is empty and the <first> tag is not mentioned or is empty, the default query or full query that pertains to the List API executes.
- If the <skip> tag is not mentioned or is empty and the <first> tag has some positive value (for example, “m”), a query that skips the “zero” row and fetches the first “m” rows executes.
- If the <skip> tag and <first> tag value are both positive (for example, “n” and “m,” respectively), a query that skips “n” rows and fetches “m” rows executes.

Suggested Use of <skip> and <first> Tags in List APIs

If a client request exceeds 8 MB of data, the server responds: “Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>.”

<Number of Rows> specifies the suggested number of rows that a single request can return to keep the data exchange under the 8-MB limit. Clients should use <Matched Rows> to determine the number of iterations that are required to retrieve the complete data set that a query tries to fetch.

Client response:

1. Client logic analyzes the server response and obtains the value of <Row Fetch>.
2. Client stores the <Row Fetch> value in a constant that depicts Row Fetch Step Size. For this example, the constant name specifies rowFetchStepSize.
3. Client executes a query for count(*) to obtain the number of rows that can be fetched from database. (Consider this step as required only if you are not using the exception to exit the loop.)
   a. Client modifies the tags <first> and <skip> in every iteration of the row fetch.

b. The first iteration starts with <skip>0<skip> and <first> rowFetchStepSize</first>.

c. Subsequent row fetch iteration have <skip>"first" tag value from previous iteration</skip> and <first>previous iteration "first" tag value + rowFetchStepSize</first>.

4. Before each iteration, client checks for the condition <skip> tag value < count(*) value. If <skip> tag value >= count(*) value, this indicates that the iteration is trying to fetch more rows than the existing number of rows in the database. In this case, break from the loop. Otherwise, the SQL Error (No Current Row) occurs, which you can use to break from loop.

Sample Code for Use of <skip> and <first> Tags in List APIs

Example 1

If the recommended RowFetch is X = 100 for ListPhoneByName, the client should use this logic:

```
StepSize = RowFetch (X = 9999)
//Two variables to store the skip and first values.
Int skip = 0;
Int first = StepSize;
//start the iteration
While (1)
{
 Try {
//A function to Modify the values of <first> and <skip> in the request with variables
values defined above
modifyRequest(FilePath);
//Sending the modified request
Response = SendExecuteSQLQuery (select SKIP <SkipCount> First <StepSize> * from endusers;)
//Modify the variables
Skip+ = StepSize;
First+= StepSize;
Check for fault_message from reply
//An SQL Exception from server while trying to fetch rows greater than that present in
database
If(fault_message .contains("No Current Row Found")){
Break;
}
Else{
Continue the loop;
}
} /* end of while*/
```

Example 2

```
#Declare variable for first and skip
long skip = 0;
long first = suggestiveRowFetch;
#Calculate number of iterations required
float precision = totalRowFetch/suggestiveRowFetch;
calculate the decimal point of variable precision
if decimal point == 0;
no. Of Iterations = totalRowFetch/suggestiveRowFetch;
if decimal point <5
no. Of Iterations = Math.round(totalRowFetch/suggestiveRowFetch)+1;
if decimal point >5
no. Of Iterations = Math.round(totalRowFetch/suggestiveRowFetch);
****Here (no. Of Iterations = 15)********
#iterator
int iLoop = 1;

while( iLoop <=no. Of Iterations){
```

```
#modify the values of first and skip in the query
String mdSQLQuery = modifySQLQuery(skip,first); /* Its like select skip 0 first 2000 *
from device for first loop*/
#send the request and get response
response = sendRequest(mdSQLQuery);
#update the skip and first variables
skip+=suggestiveRowFetch;
first+=suggestiveRowFetch;
#check for response
if(response contains MemoryConstraintException){
#undo the variable modification to get  attributes (i.e skip and first) of the query ,
that failed with an exception.
skip - =suggestiveRowFetch;
first - =suggestiveRowFetch;
wait till the requests in Progress Gets Processed;
continue the loop;

}
else{
iLoop++;
}
}
```

Sample Code for ExecuteSQLQuery

Example 1

To obtain the first X rows and then next X rows, a client should send queries as described in this section. For example, the client should use the following logic if the recommended RowFetch is X = 9999 for the query “select * from endusers;”:

```
StepSize = RowFetch (X = 9999)
SkipCount = 0
 While (1)
{
 Try {
Response = SendExecuteSQLQuery (select SKIP <SkipCount> First <StepSize> * from endusers;)
RowsReturned = Rowcount (Response);
If RowsReturned < StepSize
    Break;  /* All the rows have been fetched*/
Else
    SkipCount = SkipCount + StepSize /*Increase the SkipCount to get the next set of
rows*/
}
Catch for any exception
{
    Take appropriate action based on Exception
    Break;
}
} /* end of while*/
```

Example 2

```
#sql query to be executed
sqlQuery = "select * from device";
#send the query to server
response = sendRequest(sqlQuery);
#what is in response?
processResponse();
**** Response contains an Exception: "Query request too large.Suggestive row fetch 2000
rows.Total row fetch 30000" ****
#Declare two variables
long suggestiveRowFetch = 0; /*contains Suggestive RowFetch Count*/
```

```
long totalRowFetch = 0;      /*contains Total RowFetch*/
#fetch the two values and store it into variables.
parseExceptionMessage();
```

Testing Suggestions

This section provides tests that you can run to check various operations.

Testing ListPhoneByDescription and ListPhoneByName Methods

To test the ListPhoneByDescription and ListPhoneByName AXL methods, follow these steps:

Step 1 Populate database fields (<name>, <tkproduct>, <tkmodel>) for devices/phones to the maximum field size, which is 15 characters each.

Step 2 Populate the database with more than 60,000 devices.

Step 3 Execute the ListPhoneByDescription and ListPhoneByName AXL methods. The resulting data set has a response that is greater than 8 MB.

Expected Results: The interface returns "<API name> API request exceeds Threshold Limit. Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>." <API name> specifies the name of the API method.

Testing listDeviceByNameAndClass and listDeviceByServiceName Methods

To test the listDeviceByNameAndClass and listDeviceByServiceName AXL methods, follow these steps:

Step 1 Populate database fields (<name>, <tkproduct>, <tkmodel>) for devices/phones to the maximum field size, which is 15 characters each.

Step 2 Populate the database with more than 75,000 devices.

Step 3 Execute the listDeviceByNameAndClass and listDeviceByServiceName AXL methods. The resulting data set has a response that is greater than 8 MB.

Expected Results: The interface returns "<API name> API request exceeds Threshold Limit. Total rows matched: <Matched Rows>. Suggested row fetch: less than <Number of Rows>." <API name> specifies the name of the API method.

Testing the ExecuteSQLQuery AXL Method

To test the ExecuteSQLQuery AXL method, follow these steps:

Step 1 Populate database fields (<name>, <tkproduct>, <tkmodel>) for devices/phones to the maximum field size, which is 15 characters each.

Step 2 Create a SQL Select statement that retrieves this data.

Step 3 When the exception occurs, pick up the recommended Row Fetch Count and send the modified executeSQLQuery (in loop) by using the recommended row fetch count.

Expected Results: Demonstrate that Row Fetch Count logic works and, at the end loop, client can retrieve the entire set of required data from Cisco Unified Communications Manager.

Verifying That Tomcat Resources Are Protected

To verify that Tomcat resources are protected (Publisher server continues to operate under heavy AXL load):

Step 1 Write a script that generates a load on the AXL interface. Run executeSQLQuery to generate a response slightly less than 8 MB.

Step 2 Client runs this executeSQLQuery in a loop as soon as the transaction completes (both passed and failed transactions) for 1 hour. The client also notes the time that the request was sent, time that the response was received, and whether the response passed or failed.

Step 3 Create four instances of this script and make them run simultaneously against the same Cisco Unified Communications Manager Publisher.

Step 4 During the 1-hour load test, monitor the Tomcat JVM-related RTMT counters. In addition, use the Cisco Unified Communications Manager Administration interface to check whether you can list the devices on the system.

Step 5 At the end of the test, document the request and response times of the four clients on a timescale and note whether they succeeded or failed.

This analysis indicates whether the total responses that are processed by the AXL interface are within 16 MB. Every third concurrent request should have been rejected. By looking at the JVM on the Tomcat server that was available during the test, you can determine whether enough JVM exits to allow other applications to function properly.

Integration Considerations and Interoperability

The AXL API gives much power to developers to modify the Unified CM system database. The developer must use caution when using AXL because each API call impacts the system. Abuse of the API can lead to dropped calls and slower system performance. AXL acts as a provisioning and configuration API, not as a real-time API.

The AXL interface provides Developers with direct access to the Unified CM database via the ExecuteSQLQuery and ExecuteSQLUpdate methods. While the Dynamic Throttling mechanism protects system resources when multiple update (write) requests are received, by returning a “503: Service Unavailable” error message, there is no mechanism to guard system resources when large read requests are received.

Queries issued using the ExecuteSQLQuery method that result in a data sets greater than 8 MB may place Unified CM resources at risk. Cisco recommends developers using ExecuteSQLQuery method to follow these guidelines:

- Applications should break up all SQL queries so that the data returned is always less than 8 MB
- Use the Unified CM Data Dictionary (http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_programming_reference_guides_list.html) to help determine the maximum allowable size of any field in the database
  - ASCII characters are stored as 1-byte
  - i18n characters (UTF-8) are stored as 3-bytes

  - DB has a mix of ASCII and UTF-8 characters
- While UCMgr is processing a large query, concurrent queries should not result in data sets larger than 2 MB
- Applications should wait to receive a response before issuing subsequent queries
- Applications should not submit duplicate queries.

Note Because AXL is not a real-time API, the autologout function of extension mobility does not work when the user is logged in/out of EM via the AXL interface.

Post-Installation Steps and Troubleshooting on the Linux Platform

The system implements AXL as a Java servlet. The Java implementation provides platform independence. AXL accesses the Unified CM database by using DBL2, which is a JDBC wrapper implementation. AXL gets packaged as a WAR file. Linux RPM installs the war file for AXL on Unified CM server.

Follow the procedures in Post-Installation Steps , page 2-146, to start the AXL service and set up user permissions. Next, follow the procedures in Post-Installation Troubleshooting Checklist, page 2-147, to check the installation.

Post-Installation Steps

You can start or stop the AXL web service from Unified CM Serviceability. The service is disabled by default. You should start the service before using the AXL APIs.

Starting the AXL Service

Step 1 From the Unified CM Administration window, choose **Navigation > Unified CM Serviceability**.

Step 2 Choose **Tools > Service Activation**.

Step 3 From the **Server** box, choose the server and click **GO**.

Step 4 From **Database and Admin Services**, select **Cisco AXL Web Service** and save the changes.

Upon starting the AXL service, AXL gets deployed as a web application within Apache Tomcat. The WAR file gets deployed to Tomcat under /usr/local/thirdparty/jakarta-tomcat/webapps/axl.

Setting AXL API Access Permissions

Step 1 From the Unified CM Administration window, choose **User Management > UserGroup > Add New**.

Step 2 To add AXP API access for the new UserGroup

  a. Choose **User Management > User Group**.

b. Choose **Role > Assign Role to Group**.

c. Select **Standard AXL API Access**.

d. Click **Add Selected**.

e. On the main page, click **Save**.

Step 3 To add a user to the new UserGroup

a. Choose **User Management > User Group**.

b. Choose **UserGroup > Add End Users to Group**.

c. Select the user and click **Add Selected**.

Post-Installation Troubleshooting Checklist

Use the following checklist to avoid some common problems by fine-tuning your configuration before proceeding with the troubleshooting process:

Step 1 If the AXL client application cannot connect to the AXL service, check the following

- Is the AXL application configured with the correct IP address for the AXL server?
- Is the AXL application configured with the appropriate AXL user credentials?
- Does the application server have HTTPS connectivity to the AXL server?

  Use this URL for accessing AXL: https://server-name:port/axl/ (port is 8443).
- Is HTTPS (secure) configured for AXL?

Step 2 Verify basic AXL functionality by performing the procedure that follows:

1. Go to the AXL API URL via a web browser.

   For instance, enter https://server-name:8443/axl/ in the address text box.
2. When prompted for user name and password, use the standard administrator login, or use the user name that is associated with a user group that is assigned the AXL role.
3. Look for a plain page that states the AXL listener is working and accepting requests but only communicates via POST**.**

This procedure verifies functionality and user access.

Step 3 If the AXL functions or requests are failing with error as User Authorization error: “Access to the requested resource has been denied,” check whether the user has the permission to the Standard AXL API Access. You can check this from the Permission Information section of the EndUser configuration window.

Step 4 If the AXL functions or requests are failing, check the following:

- AXL logs for AXL or SOAP error responses. See the “AXL Error Codes” section on page 2-163.
- For further debugging, you can view the AXL log files with the RTMT application.

Step 5 Check that applications in a cluster configuration are connected to the AXL service only on the Unified CM Publisher server if the application needs to modify the database.

AXL Trace Logs

AXL trace logs contain the text of every AXL request and response, along with user and origination IP information. Trace logs prove useful for identifying who is making AXL requests, inspecting the AXL XML request for format or syntax errors, and determining the actual AXL service response or errors.

The system writes the AXL trace logs to the /var/log/active/tomcat/logs/axl/log4j directory. You can view them with RTMT. File names are axl####.log, where # represents a number from 0000 (zero) to the maximum number of files allowed. The maximum file size is 1 MB by default. The maximum number of stored files defaults to 10. You can change these settings through the Serviceability windows .

Note If an AXL login request contains a <password> tag with an xmlns attribute value, versions of the AXL API prior to 6.0(1) or 5.1(2) log the password in clear text. In later versions of the API, the system replaces the password with an "*" character.

For .NET WSDL applications, including the xmlns attribute in the <password> tag would be typical behavior, and, in earlier releases, this could represent a security issue. If the SOAP message body contains a namespace tag, you do not need to specify the xmlns attribute for each individual tag.

While analyzing the log files:

- Determine which log file is currently active by timestamp.
- Look for Exception traces that indicate processing errors.
- If no traces are being added, verify that Tomcat is running, AXL is currently activated, and a client application is attempting to communicate with the AXL API.

The following sample shows the AXL trace log output:

```
2007-03-17 05:32:26,512 INFO  [http-8443-Processor21] axl.AxlListener - Received request
1173323669700 from CCMAdministrator at IP 10.77.31.203
2007-03-17 05:32:26,513 INFO  [http-8443-Processor21] axl.AxlListener - <!-- edited with
XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Jerry Vander Voord (Cisco Systems)
--><SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Body>
<axlapi:addGatekeeper sequence="1" xmlns:axlapi="http://www.cisco.com/AXL/API/7.0"
xmlns:axl="http://www.cisco.com/AXL/7.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cisco.com/AXL/API/7.0 axlsoap.xsd">
<gatekeeper>
    <name>AXL-Sample-GK1</name>
    <description>This is a sample gatekeeper</description>
    <rrqTimeToLive>30</rrqTimeToLive>
    <retryTimeout>30</retryTimeout>
    <enableDevice>false</enableDevice>
</gatekeeper>
</axlapi:addGatekeeper></SOAP-ENV:Body></SOAP-ENV:Envelope>
2007-03-17 05:32:26,668 INFO  [http-8443-Processor21] axl.Handler - Handler initializing
2007-03-17 05:32:26,788 INFO  [http-8443-Processor21] axl.AxlListener - <?xml
version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Header/><SOAP
-ENV:Body>
<axl:addGatekeeperResponse xmlns:axl=http://www.cisco.com/AXL/7.0
xmlns:xsi="http://www.cisco.com/AXL/7.0" sequence="1">
<return>{7EB31958-C24A-3E63-3E4B-A8446365D35}</return>
</axl:addGatekeeperResponse></SOAP-ENV:Body></SOAP-ENV:Envelope>
2007-03-17 05:32:26,789 INFO  [http-8443-Processor21] axl.AxlListener - Request
1173323669700 was process in 356ms
```

The AXL trace log contains:

- Time that the request was received - 05:32:26,512
- Client IP address - IP 10.77.31.203
- Client user ID - CCMAdministrator
- Request ID number - 1173323669700
- Request contents – addGatekeeper, <gatekeeper>, <name>, and so on
- Response contents - <name>
- Time taken for the response - 356 ms

Follow these steps to set the AXL trace level from the Serviceability window:

Step 1 From the Unified CM Administration window, choose **Application > Unified CM Serviceability**.

Step 2 Choose **Trace > Configuration**.

Step 3 From the **Servers** column, select the server and click **GO**.

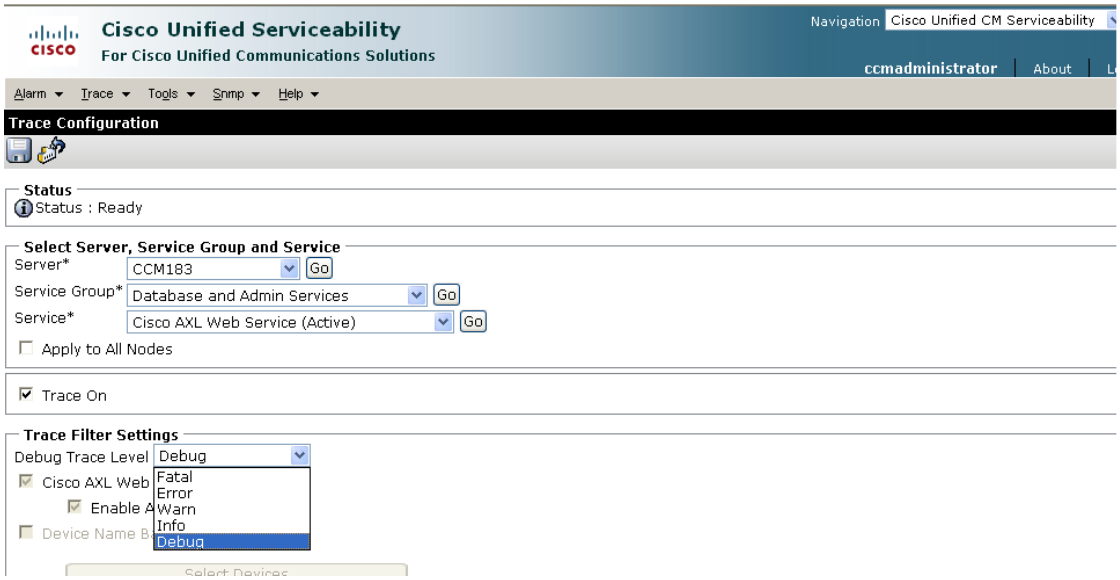
Step 4 From the **Service Group** box, select **Database And Admin Services** and click **GO.**

Step 5 From the **Services** box, select the **Cisco AXL Web Service** and click **GO**.

Step 6 Check the **Trace On** check box.

Step 7 If you want the trace to apply to all Unified CM servers in the cluster, select the **Apply to All Nodes** check box.

Step 8 From the **Debug Trace Level** field, select **Debug**.



Step 9 You can set trace output options from the same window.

Note You should enable AXL logs only on request from Cisco TAC or Cisco Developer Services.

Using the AXL API with AXIS

This section explains how to use the AXLAPI.wsdl file in a Java programming environment.

Cisco has verified the AXLAPI.wsdl file in the WSDL-AXIS folder in the AXIS environment.

Cisco provides the associated schema file, AXLSoap.xsd, only for code generation. Cisco has modified this file to minimize the backwards compatibility impact of future changes in the database schema. Use the WSDL and the schema files in the parent directory for reference and for validation of responses.

The WSDL in Unified CM release 8.0(1) onwards is compatible with AXIS 2 and AXIS 1.4.

When you run the wsdl2Java utility to create the Java source code by using AXLAPI.wsdl, the utility throws two errors that are specific to AXIS_1_4. For further details on these errors, refer to http://issues.apache.org/jira/browse/AXIS-2545 and http://issues.apache.org/jira/browse/AXIS-1280.

The incorrect ordering of the parameters that are passed to the constructor causes the first AXIS jira error. A code example follows:

```
public class XNPDirectoryNumberShareLineAppearanceCSS  extends
com.cisco.www.AXL.API._7_0.XCallingSearchSpace  implements java.io.Serializable {
>
>
super(
            uuid,
            name,
            description,
            clause,
            dialPlanWizardGenId,
            members);
```

However, the parent constructor is defined as:

```
public XCallingSearchSpace(
           org.apache.axis.types.Name name,
           java.lang.String description,
           java.lang.String clause,
           org.apache.axis.types.NonNegativeInteger dialPlanWizardGenId,
           com.cisco.www.AXL.API._7_0.XCallingSearchSpaceMember[] members,
           java.lang.String uuid) {
           this.name = name;
           this.description = description;
           this.clause = clause;
           this.dialPlanWizardGenId = dialPlanWizardGenId;
           this.members = members;
           this.uuid = uuid;
    }
```

You need to change either the constructor or the constructor calling as shown below:

```
super(
            name,
            description,
            clause,
            dialPlanWizardGenId,
            members,
        uuid);
```

The second AXIS jira error relates to having a string constructor for simple types; for example

```
// Simple Types must have a String constructor
    public XLoadInformation(java.lang.String _value) {
    super(_value);
}
```

For such cases, the corresponding schema file (axl.xsd) in the parent schema folder must be referred and must implement the string class that these classes can inherit.

Using the AXL API in a .NET Environment

To integrate the AXL API with a .NET client, you must modify the Cisco-provided WSDL and XSD files.

Cisco provides the associated schema file, AXLSoap.xsd, only for code generation. Cisco has modified this file to minimize the backwards compatibility impact of future changes in the database schema. Use the WSDL and the schema files in the parent directory for reference and for validation of responses.

The inability of .NET to handle complex schemas necessitates some of the changes that are described below.

Required Changes to the Generated Code

After running WSDL.exe, you must make several changes to the generated code. Run WSDL.exe by using the following command:

```
wsdl.exe AXLAPI.wsdl axlsoap.xsd
```

This command generates the file AXLAPIService.cs. The class AXLAPIService in AXLAPIService.cs requires at least three changes:

1. Create an ICertificatePolicy-derived class, which will later be associated with our service. This class represents a brute-force approach to policy and certificate management. You need to use this method in 5.x and 6.x AXL due to the use of HTTPS.

```
public class BruteForcePolicy : System.Net.ICertificatePolicy
{
    public bool CheckValidationResult(System.Net.ServicePoint sp,
            System.Security.Cryptography.X509Certificates.X509Certificate cert,
            System.Net.WebRequest request, int problem)
    {
    return true;
    }
}
```

2. Modify the service constructor to take username/password credentials, with the Unified CM IP address as an argument, and associate the BruteForcePolicy class with the static CertificatePolicy manager.

```
public AXLAPIService(string ccmIp, string user, string password)
{
    System.Net.ServicePointManager.CertificatePolicy = new BruteForcePolicy();

    this.Url = "https://" + ccmIp + ":8443/axl/";
    this.Credentials = new System.Net.NetworkCredential(user, password);
}
```

3. The .NET framework uses the *expects* header differently (http://issues.apache.org/bugzilla/show_bug.cgi?id=31567). Several possible workarounds exist to this problem:
   a. Override the GetWebRequest method to use HTTP 1.0 due to an error between TOMCAT/AXIS and the .NET HTTP 1.1 Web Service request mechanism.

```
        protected override System.Net.WebRequest GetWebRequest(Uri uri)
        {
            System.Net.HttpWebRequest request = base.GetWebRequest (uri) as
                System.Net.HttpWebRequest;
            request.ProtocolVersion = System.Net.HttpVersion.Version10;

            return request;
        }
```

b. Override the GetWebRequest method to manually embed the authentication string. If you do this, do not use the line

```
this.Credentials = new System.Net.NetworkCredential(user, password);
```

from the constructor that is provided in point 2 earlier in this section.

```
protected override System.Net.WebRequest GetWebRequest(Uri uri)
{
    System.Net.HttpWebRequest request =(System.Net.HttpWebRequest)base.GetWebRequest(uri);
    if (this.PreAuthenticate)
    {
        System.Net.NetworkCredential nc = this.Credentials.GetCredential(uri,"Basic");
        if (nc != null)
        {
            byte[] credBuf = new System.Text.UTF8Encoding().GetBytes(nc.UserName + ":" +
nc.Password);
            request.Headers["Authorization"] = "Basic " + Convert.ToBase64String(credBuf);
        }
    }
    return request;
}
```

c. If you use wsdl2wse (WSE library) instead of wsdl.exe, you cannot override the HTTP version or supply HTTP headers manually. To use WSE, you must set the keepalive header to false for the generated class, or set the user-agent to restricted. This technique will work as an alternative to steps a and b.

Backward Compatibility Issues

When you add the definitions for new Cisco Unified IP Phone devices to the Unified CM database, the original WSDL that was sent out for that Unified CM becomes outdated. For example, the XModel enumeration in Cisco CallManager Release 4.1.3 does not contain the Cisco Unified IP Phone 7961G-GE.

However, if you install the latest device pack that contains that device information into your release 4.1.3 environment, that value will be returned if you use the listAllDevices or getPhone commands for that device name. This causes .NET to throw an exception when it encounters the new model because the definition does not contain the mode.

More generally, almost all enumerations in AXLEnums.xsd could change in some future release, which in turn might create backward incompatibility with your code. To address this issue, Cisco has changed the type of all of the tags that use any of these enumerations to String and added an annotation to that tag that specifies where to look for the correct value (AXLEnums.xsd).

Tag Serialization Issues

If you generate the client stub by using wsdl.exe, you may find that some fields that have default values that are defined in the schema would not work if passed in the AXL request. For example, in the updatePhoneReq class of the generated client stub, a field named "ignorePresentationIndicators" has a default value of "False" defined in the schema.

```
[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.cisco.com/AXL/7.0")]
    public class UpdatePhoneReq : APIRequest {
         .
         .        .

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unquali
fied)]
         [System.ComponentModel.DefaultValueAttribute(false)]
         public bool ignorePresentationIndicators = false;
         .
         .
    }
```

When this tag is sent with a value of false, XmlSerializer does not serialize this tag because of a design restriction in Microsoft .NET Framework 1.0. Refer to http://support.microsoft.com/kb/325691.

To work around this problem, comment out all instances of

```
[System.ComponentModel.DefaultValueAttribute(XXX)]
```

in the generated client stub as shown:

```
[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.cisco.com/AXL/7.0")]
public class UpdatePhoneReq : APIRequest {
                .
                .                .
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unquali
fied)]
        // Comment this line below
        //[System.ComponentModel.DefaultValueAttribute(false)]
        public bool ignorePresentationIndicators = false;                .
                .
         }
```

A second issue that is found when you are using the version of wsdl.exe that comes with .NET 1.0 is that some tags, including fkcallingsearchspace_autoregistration, do not get updated to null/none in the database.

This appears to be an issue in which .NET does not serialize tags that are defined as nillable=true in the schema.

For example, to work around this limitation for the tag callingSearchSpace in updatePhoneReq in the generated stub, you can remove the "Form=System.Xml.Schema.XmlSchemaForm.Unqualified" from

```
     [System.Xml.Serialization.XmlElementAttribute("name", typeof(string),
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
          [System.Xml.Serialization.XmlElementAttribute("uuid", typeof(string),
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
          [System.Xml.Serialization.XmlChoiceIdentifierAttribute("ItemElementName")]
          public string Item;
```

With this change, the serializer will serialize the tags. Passing the tag value as "" will set the callingSearchSpace to null/None. The same workaround applies to other such tags.

Names Containing Special Characters

Using the version of wsdl.exe that comes with .NET 1.0, Cisco has found that when attempting to add elements like gatewayEndpoint,MGCPEndpoint or CSS where the name contains special characters, the elements do not get updated in the database properly.

For example, a gatewayEndpoint with name="AALN@SAA000011114444" sent as name="AALN_x0040_SAA000011114444" in the AXL request.

This appears to be a limitation in .NET serialization of tags that are defined as type xsd:Name in the schema.

In the XML specification, the type xsd:name is defined as a token that begins with a letter or one of a few punctuation characters and continues with letters, digits, hyphens, underscores, colons, or periods, together known as name characters. Thus, xsd:name does not allow any special characters such as '@' or '/'.

One workaround involves changing the data type from "Name" to "string" in the generated stub:

Original Code

```
public class XDevice {
    [System.Xml.Serialization.XmlElementAttribute
    (Form=System.Xml.Schema.XmlSchemaForm.Unqualified,DataType="Name")]
    public string name;
```

Modified Code

```
public class XDevice {
    [System.Xml.Serialization.XmlElementAttribute(typeof(string),
        Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
    public string name;
```

With this modification, the special characters in the name will be updated in the database without any conversion.

Returned Namespace for AXIS and .NET Applications

By default, the AXLAPI.wsdl carries the namespace http://www.cisco.com/AXL/API/7.0. The generated client stubs also have this namespace. In some situations, you must change the namespace in AXLAPI.wsdl before creating the client stubs.

The namespace that is returned in the AXL response depends on two factors:

1. Whether the SOAPAction attribute in the HTTP header had the value “CUCM:DB ver=7.0.”
2. The value of the “Send Valid Namespace in AXL Response” service parameter in the Cisco Unified Communications Manager Administration Service Parameter window.

If the SOAPAction attribute in the HTTP header has the value “CUCM:DB ver=7.0”:

- The AXL response will have the namespace value that the “Send Valid Namespace in AXL Response” service parameter specifies: either http://www.cisco.com/AXL/API/7.0 or http://www.cisco.com/AXL/7.0.

- If you set the service parameter “Send Valid Namespace in AXL Response” to true, the namespace that is returned in the AXL response will be http://www.cisco.com/AXL/API/7.0, which will match the namespace that is specified in AXLAPI.wsdl.
- If you set this service parameter to False, the namespace that is returned in the AXL response will be http://www.cisco.com/AXL/7.0.

If the SOAPAction attribute has any other value, the AXL response will have the namespace http://www.cisco.com/AXL/API/1.0 or http://www.cisco.com/AXL/1.0, depending on the value of the service parameter.

Example AXL Requests

No platform considerations exist in Unified CM Release 7.0(1). The client must be able to send an HTTPS request to the AXL endpoint.

The following examples describe how to make an AXL request and read back the response to the request.

Ensure each SOAP request is sent to the web server via an HTTPS POST. The endpoint URL represents the AXL web service that is running on a Unified CM server. The following list contains the only four required HTTPS headers:

- POST :8443/axl/

  The first header specifies that this particular POST is intended for the Cisco AXL Web Service. The AXL API only responds to the POST method.

- content-type: text/xml

  The second header confirms that the data that is being sent to AXL is XML. If this header is not found, the system returns an HTTP 415 error to the client.

- Authorization: Basic <some Base 64 encoded string>

  The third header gives the Base64 encoding of the user name and password for the administrator of the AXL Server. Because Base64 encoding takes three 8-bit bytes and represents them as four printable ASCII characters, if the encoded header does not contain an even multiple of four ASCII characters (16, 20, 24, and so on), you must add padding characters (=) to complete the final group of four as in the following examples.

  If authentication of the user fails, the system returns an HTTP 401 Access Denied error to the client.

- content-length: <a positive integer>

  The fourth header specifies the length (in bytes) of the AXL request.

Note If a request that is greater than 40 kilobytes is received, the system returns an HTTP 413 error message.

The following example contains an HTTPS header for an AXL SOAP request:

```
POST :8443/axl/
Host: axl.myhost.com:8443
Accept: text/*
Authorization: Basic bGFycnk6Y3VybHkgYW5kIG1vZQ==
Content-type: text/xml
SOAPAction: "CUCM:DB ver=7.0"
Content-length: 613
```

The following AXL request gets used in the code examples that display in the following sections. This example shows a getPhone request:

```
POST :8443/axl/
Host: axl.myhost.com:8443
Accept: text/*
Authorization: Basic bGFycnk6Y3VybHkgYW5kIG1vZQ==
Content-type: text/xml
SOAPAction: "CUCM:DB ver=7.0"
Content-length: 613

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <SOAP-ENV:Body>
        <axl:getPhone xmlns:axl="http://www.cisco.com/AXL/7.0"
xsi:schemaLocation="http://www.cisco.com/AXL/7.0 http://ccmserver/schema/axlsoap.xsd
"  sequence="1234">
            <phoneName>SEP222222222245</phoneName>
        </axl:getPhone>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

C or C++ Example

This code example uses a hard-coded AXL request and sends it to the AXL server that is running on the local system (localhost). It then reads the response and outputs the response to the screen:

```
#include <sys/socket.h>
#include <sys/types.h>
#include <stdlib.h>
#include <openssl/ssl.h>
#include <stdio.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <strings.h>
#include <openssl/x509.h>
#include <openssl/crypto.h>
#include <iostream>
#include <string>
using namespace std;
typedef unsigned char byte;



void encodeBase64( const string& inBuf, string &outBuf )
{
    unsigned int i;
    unsigned int j;
    bool hiteof = false;
    byte dtable[256];

    outBuf.erase();

    for(i= 0;i<9;i++)
    {
       dtable[i]= 'A'+i;
       dtable[i+9]= 'J'+i;
       dtable[26+i]= 'a'+i;
       dtable[26+i+9]= 'j'+i;
    }
    for(i= 0;i<8;i++)
    {
       dtable[i+18]= 'S'+i;
       dtable[26+i+18]= 's'+i;
    }
```

```
    for(i= 0;i<10;i++)
    {
       dtable[52+i]= '0'+i;
    }

    dtable[62]= '+';
    dtable[63]= '/';

    j = 0;
    while(!hiteof)
    {
       byte igroup[3],ogroup[4];
       int c,n;

       igroup[0]= igroup[1]= igroup[2]= 0;
       for(n= 0;n<3;n++){
          if( j < inBuf.size() )
          {
             c = inBuf[j++];
          } else
          {
             hiteof = true;
             break;
          }
          igroup[n]= (byte)c;
       }
       if(n> 0)
       {
          ogroup[0]= dtable[igroup[0]>>2];
          ogroup[1]= dtable[((igroup[0]&3)<<4)|(igroup[1]>>4)];
          ogroup[2]= dtable[((igroup[1]&0xF)<<2)|(igroup[2]>>6)];
          ogroup[3]= dtable[igroup[2]&0x3F];

          if(n<3)
          {
             ogroup[3]= '=';
             if(n<2)
             {
                ogroup[2]= '=';
             }
          }
          for(i= 0;i<4;i++)
          {
             outBuf += ogroup[i];
          }
       }
    }
}

string getAuthorization()
{
  string m_encode64,name;
  //You should change name to your own axl server user name and passwd
  //in this example, "CCMAdministrator" is the user name and "cisco_cisco" is the passwd.
  name="CCMAdministrator:cisco_cisco";
  encodeBase64(name,m_encode64);
  return m_encode64;
}

void
BuildDeviceNameSQL(string &buf,  // Buffer to build AXL
                                  string& deviceNumber, // DN
                                  string& seqNum )
{
    const int BUFSIZE = 2048;
    char buff[BUFSIZE];           // Temp buffer
    string strHTTPHeader;         // HTTP/AXL Header
    string strAXLRequest;         // AXL Request
```

```
    strAXLRequest = "<SOAP-ENV:Envelope xmlns:SOAP-ENV=";
    strAXLRequest += "\"http://schemas.xmlsoap.org/soap/envelope/\"";
    strAXLRequest += " xmlns:SOAP-ENC=\"http://schemas.xmlsoap.org/soap/encoding/\"";
    strAXLRequest += " xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"";
    strAXLRequest += " xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"> ";
    strAXLRequest += "<SOAP-ENV:Body> ";
    strAXLRequest += "<m:executeSQLQuery xmlns:m=\"http://www.cisco.com/AXL/API/7.0\"
                          sequence=\"" + seqNum + "\"> ";
    strAXLRequest += "<m:sql> ";
    strAXLRequest += "SELECT * FROM Device ";
    strAXLRequest += "</m:sql> ";
    strAXLRequest += "</m:executeSQLQuery> ";
    strAXLRequest += "</SOAP-ENV:Body> ";
    strAXLRequest += "</SOAP-ENV:Envelope>";

    strHTTPHeader = "POST /axl/ HTTP/1.1\r\n";

    strHTTPHeader += "Host: localhost:8443\r\n";
    strHTTPHeader += "Accept: text/*\r\n";
    strHTTPHeader += "Authorization: Basic ";
    strHTTPHeader += getAuthorization() + "\r\n";
    strHTTPHeader += "Content-type: text/xml\r\n";
    strHTTPHeader += "SOAPAction: \"CUCM:DB ver=7.0\"\r\n";
    strHTTPHeader += "Content-length: ";

    // temporarily use the buffer to store the length of the request
    sprintf( buff, "%d", strAXLRequest.length() );

    strHTTPHeader += buff;
    strHTTPHeader += "\r\nConnection: Keep-Alive";
    strHTTPHeader += "\r\n\r\n";

    // put the HTTP header and SOAP XML together
    buf = strHTTPHeader + strAXLRequest;

    return;

}

int main(int argc, char** argv)
{
  struct sockaddr_in saddr;
  SSL_METHOD *meth;
  SSL_CTX *sslctx;
  SSL *ssl;
  X509* server_cert;
  string buff,line,seqnum;
  char buffer[2048];
  int status,error;
  char *str;

  if( argc!=3 )
  {
     printf("Usage : ssltest <ip> <port> \n");
     printf("Usage : the default port is 8443 \n");
     printf("Usage : the ip is the ip of ccm5.0 \n");
     printf("Example: ssltest 10.77.31.168 8443 \n");

     exit(2);
  }

  int sock=socket(AF_INET,SOCK_STREAM,IPPROTO_TCP);
  if(sock<0)
  {
    printf("create socket failed\n");
    exit(1);
```

```
}
saddr.sin_family=AF_INET;
saddr.sin_port=htons(atoi(argv[2]));
saddr.sin_addr.s_addr =inet_addr(argv[1]);
status=connect(sock,(struct sockaddr *)&saddr,sizeof(saddr));
if(status<0)
{
  printf("connect to %s failed\n",argv[1]);
  exit(2);
}
SSL_library_init();
meth=TLSv1_client_method();
sslctx=SSL_CTX_new(meth);
if(!sslctx)
{
   printf("SSL_CTX_new failed\n");
   close(sock);
   exit(3);
}
SSL_CTX_set_verify(sslctx,SSL_VERIFY_NONE,NULL);
ssl =SSL_new(sslctx);
if(!ssl)
{
  printf("SSL_new failed\n");
  close(sock);
  exit(4);
}
status=SSL_set_fd(ssl,sock);
if(!status)
{
  printf("SSL_set_fd failed\n");
  close(sock);
  exit(5);
}
SSL_set_mode(ssl,SSL_MODE_AUTO_RETRY);
status=SSL_connect(ssl);
error=SSL_get_error(ssl,status);
switch(error)
{
  case SSL_ERROR_NONE:
    printf("connect successful\n");
    break;
  case SSL_ERROR_ZERO_RETURN:
    printf("peer close ssl connection \n");
    break;
  default:
    printf("connect error is %d\n",error);
 }

 server_cert = SSL_get_peer_certificate (ssl);
 if(!server_cert)
 {
   printf("get server certificate failed!\n");
   SSL_shutdown(ssl);
   close(sock);
   exit(6);
 }
 str= X509_NAME_oneline(X509_get_subject_name (server_cert),0,0);
 if(str)
 {
   printf("subject :%s\n",str);
 }
 else
   printf("subject is empty\n");
 str = X509_NAME_oneline (X509_get_issuer_name  (server_cert),0,0);
 if(!str)
   printf("issuer name is :%s\n",str);
```

```
   else
     printf("issuer name is empty \n");
   line="12";
   seqnum="1234";
   BuildDeviceNameSQL(buff,line,seqnum);
   SSL_write(ssl,buff.c_str(),buff.length());
   printf("\n");
   printf("\n");
   printf("Request sent is:\n");
   printf(buff.c_str());
   printf("\n");
   printf("\n");
   SSL_read(ssl,buffer,sizeof(buffer));

   printf("Response from server is: \n%s\n",buffer);
   status=SSL_shutdown(ssl);
   if(status==1)
     printf("shutdown successful\n");
   else
     printf("\nshutdown error code is %d\n",status);
   close(sock);
}
```

Java Example

This code example uses a hard-coded AXL request and sends it to the AXL server that is running on the local system (localhost). It then reads the response and outputs the response to the screen.

```
import java.io.*;
import java.net.*;
import javax.net.ssl.*;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;

public class AXLJavaClient {
    public static void main(String[] args) {
        byte[] bArray = null; // buffer for reading response from
        Socket socket = null; // socket to AXL server
        OutputStream out = null; // output stream to server
        InputStream in = null; // input stream from server

        String sAXLSOAPRequest = "";
        // HTTPS header and SOAP payload
        String sAXLRequest = null; // will hold only the SOAP payload
        //username=CCMAdministrator and password=cisco_cisco
        String authorization = "CCMAdministrator" + ":" + "cisco_cisco";
        // base64 encoding of the username and password
        authorization = new sun.misc.BASE64Encoder().encode(authorization.getBytes());
        // Form the http header
        sAXLSOAPRequest = "POST /axl/ HTTP/1.0\r\n";
        sAXLSOAPRequest += "Host:localhost:8443\r\n";
        sAXLSOAPRequest += "Authorization: Basic " + authorization + "\r\n";
        sAXLSOAPRequest += "Accept: text/*\r\n";
        sAXLSOAPRequest += "Content-type: text/xml\r\n";
        sAXLSOAPRequest += "SOAPAction: \"CUCM:DB ver=7.0\"\r\n";
        sAXLSOAPRequest += "Content-length: ";

        // Build the SOAP payload
        sAXLRequest = "<SOAP-ENV:Envelope
xmlns:SOAP-ENV=\"http://schemas.xmlsoap.org/soap/envelope/\" ";
        sAXLRequest += "xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"> ";
```

```
        sAXLRequest += "<SOAP-ENV:Body> <axl:getPhone
xmlns:axl=\"http://www.cisco.com/AXL/7.0\" ";
        sAXLRequest += " xsi:schemaLocation=\"http://www.cisco.com/AXL/7.0 http://
ccmserver/schema/axlsoap.xsd\" ";
        sAXLRequest += "sequence=\"1234\"> <phoneName>SEP000000000009</phoneName>";
        sAXLRequest += "</axl:getPhone> </SOAP-ENV:Body> </SOAP-ENV:Envelope>";


        // finish the HTTPS Header
        sAXLSOAPRequest += sAXLRequest.length();
        sAXLSOAPRequest += "\r\n\r\n";

        // now add the SOAP payload to the HTTPS header, which completes the AXL
        // SOAP request
        sAXLSOAPRequest += sAXLRequest;
        try {
            AXLJavaClient axl = new AXLJavaClient();
        // Implement the certificate-related stuffs required for sending request via https
            X509TrustManager xtm = axl.new MyTrustManager();
            TrustManager[] mytm = { xtm };
            SSLContext ctx = SSLContext.getInstance("SSL");
            ctx.init(null, mytm, null);
            SSLSocketFactory sslFact = (SSLSocketFactory) ctx.getSocketFactory();
            socket = (SSLSocket) sslFact.createSocket("10.77.31.203",
Integer.parseInt("8443"));
            in = socket.getInputStream();
            // send the request to the server
            // read the response from the server
            StringBuffer sb = new StringBuffer(2048);
            bArray = new byte[2048];
            int ch = 0;
            int sum = 0;
            out = socket.getOutputStream();
            out.write(sAXLSOAPRequest.getBytes());
            while ((ch = in.read(bArray)) != -1) {sum += ch;sb.append(new String(bArray,
0, ch));
            }
            socket.close();
            // output the response to the standard output
            System.out.println(sb.toString());
        } catch (UnknownHostException e) {
            System.err.println("Error connecting to host: " + e.getMessage());
            return;
        } catch (IOException ioe) {
            System.err.println("Error sending/receiving from server: " +
ioe.getMessage());
            // close the socket
        } catch (Exception ea) {
            System.err.println("Unknown exception " + ea.getMessage());
            return;
        }
        finally{
            try {
                if (socket != null)
                    socket.close();
            } catch (Exception exc) {
                    exc.printStackTrace();
                    System.err.println("Error closing connection to server: "+
exc.getMessage());
            }
        }
    }

    public class MyTrustManager implements X509TrustManager {
```

```
MyTrustManager() {
    // create/load keystore

}

public void checkClientTrusted(X509Certificate chain[], String authType)
        throws CertificateException {

}

public void checkServerTrusted(X509Certificate chain[], String authType)
        throws CertificateException {

}

public X509Certificate[] getAcceptedIssuers() {

    return null;
```

In addition to these examples, refer to the AXL SQL Toolkit, which is available for download from the Unified CM server at *https://ccmserver:8443/plugins/axlsqltoolkit.zip*.

Using executeSQLUpdate

This example illustrates the use of the executeSQLUpdate request:

```
POST :8443/axl/
Host: axl.myhost.com:8443
Accept: text/*
Authorization: Basic bGFycnk6Y3VybHkgYW5kIG1vZQ==
Content-type: text/xml
SOAPAction: "CUCM:DB ver=7.0"
Content-length: 613

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body>
        <axlapi:executeSQLUpdate sequence="1"
xmlns:axlapi="http://www.cisco.com/AXL/API/7.0" xmlns:axl="http://www.cisco.com/AXL/7.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cisco.com/AXL/API/7.0 axlsoap.xsd">
            <sql>
                insert into device (fkPhoneTemplate,fkDevicePool,tkclass, tkpreemption,
tkdeviceprofile,  tkmodel, tkdeviceprotocol, tkproduct, description,
tkstatus_mlppindicationstatus, name, pkid) values ('Standard 7941 SCCP','default',1, 2, 2,
115, 0, 115, '', 0, 'Cisco 7941', newid())
            </sql>
        </axlapi:executeSQLUpdate>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Using executeSQLQuery

This example illustrates the use of the executeSQLQuery request:

```
POST :8443/axl/
Host: axl.myhost.com:8443
Accept: text/*
Authorization: Basic bGFycnk6Y3VybHkgYW5kIG1vZQ==
Content-type: text/xml
```

```
SOAPAction: "CUCM:DB ver=7.0"
Content-length: 613

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body>
        <axlapi:executeSQLQuery sequence="1"
xmlns:axlapi="http://www.cisco.com/AXL/API/7.0"
xmlns:axl="http://www.cisco.com/AXL/API/7.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cisco.com/AXL/API/7.0 axlsoap.xsd">
            <sql>SELECT * from numplan</sql>
        </axlapi:executeSQLQuery>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

AXL Error Codes

If an exception occurs on the server, or if any other error occurs during the processing of an AXL request, the system returns an error in the form of a SOAP Fault message. For example:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
            <faultcode>SOAP-ENV:Client</faultcode>
            <faultstring>Device not found with name SEP003094C39708.</faultstring>
            <detail xmlns:axl="http://www.cisco.com/AXL/7.0"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="http://www.cisco.com/AXL/API/7.0
            http://myhost/CCMApi/AXL/V1/axlsoap.xsd">
            <axl:error sequence="1234">
            <code>0</code>
            <message>
            <![CDATA[
            Device not found with name SEP003094C39708.
            ]]>
            </message>
            <request>doDeviceLogin</request>
            </axl:error>
            </detail>
        </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Fault messages can also contain more detailed information. The following example depicts a detailed SOAP Fault:

```
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
            <faultcode>SOAP-ENV:Client</faultcode>
            <faultstring>Device not found with name SEP003094C39708.</faultstring>
            <detail  xmlns:axl="http://www.cisco.com/AXL/7.0"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://www.cisco.com/AXL/7.0
                    http://myhost/CCMApi/AXL/V1/axlsoap.xsd">
                <axl:error  sequence="1234">
                    <code>0</code>
                    <message>
```

```
<![CDATA[
Device not found with name SEP003094C39708.
]]>
                    </message>
                    <request>doDeviceLogin</request>
                </axl:error>
            </detail>
        </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The <detail> element of a SOAP Fault includes error codes. The axl:Error elements represent the errors. If a response to a request contains an <error> element, the user agent can determine the cause of the error by looking at the subelements of the <error> tag.

The user agent uses the <code> element, a numerical value, to find what type of error occurred.

The following table explains the error codes that the <code> tag might have:

Error Code	Description
5000	**Unknown Error**—An unknown error occurred while the request was processed. This can occur due to a problem on the server but can also be due to errors in the request.
5002	**Unknown Request Error**—The user agent submitted a request that is unknown to the API.
5003	**Invalid Value Exception**—The API detected an invalid value in the XML request.
5007	**Item Not Valid Error**—The system identified the specified item as invalid, which means that it does not exist or that it was specified incorrectly at input.

message

The system provides the <message> element, so the user agent gets a detailed error message that explains the error.

request

The system provides the <request> element, so the user agent can determine the type of request that generated this error. Because this element is optional, it may not always appear.

CHAPTER 3

Administrative XML Operations by Release

Table 3-1 lists new, changed, and deprecated Administrative XML (AXL) operations by release. It may also list operations that are under consideration or review (UCR). Operation details can be found in Chapter 2, "Administrative XML Programming".

Table legend:

- —Supported
- —Not supported
- —Modified

Operations By Release

Table 3-1 lists new, changed, and deprecated Administrative XML (AXL) operations in Unified CM release 8.6(1) andthe earlier releases.

Table 3-1 Operations by Release

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
AARGroup (AarGroup in Unified CM 8.0 onwards)	addAARGroup	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	getAARGroup	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	listAarGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeAARGroup	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
	updateAARGroup	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
AARGroupByName (use listAarGroup API in Unified CM 8.0 onwards)	listAARGroupByName	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗
AARGroupMatrix (updateAarGroupMatrix in Unified CM 8.0 onwards)	updateAARGroupMatrix	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
AllProcessNodes	listAllProcessNodes	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ApplicationToSoftkeyTemplate	addApplicationToSoftkeyTemplate	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	removeApplicationToSoftkeyTemplate	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
ApplicationUser	addApplicationUser	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	getApplicationUser	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	removeApplicationUser	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	updateApplicationUser	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
AppUser	addAppUser	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getAppUser	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listAppUser	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeAppUser	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateAppUser	✓	✓	✓	ℹ	✓	✓	✓	✓	✓	✓
AttendantConsoleHuntGroup	addAttendantConsoleHuntGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	getAttendantConsoleHuntGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	removeAttendantConsoleHuntGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateAttendantConsoleHuntGroup	ℹ	✓	✓	✓	✓	✓	✓	✓	✓	✓
AttendantConsoleUser	addAttendantConsoleUser	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	getAttendantConsoleUser	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	listAttendantConsoleUser	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeAttendantConsoleUser	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateAttendantConsoleUser	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AuthenticateUser	doAuthenticateUser	✓	✓	✓	✓	✓	✓	✓	✓	ℹ	✓
CalledPartyTransformationPattern	addCalledPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	getCalledPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	listCalledPartyTransformationPattern	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCalledPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	updateCalledPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
CallerFilterList	addCallerFilterList	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	getCallerFilterList	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	listCallerFilterList	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallerFilterList	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateCallerFilterList	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
CallingPartyTransformationPattern	addCallingPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	getCallingPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	listCallingPartyTransformationPattern	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallingPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	updateCallingPartyTransformationPattern	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
CallIntercept	addCallIntercept	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCallIntercept	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCallIntercept	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallIntercept	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCallIntercept	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CallManager (add and remove APIs are deprecated in Unified CM 8.0)	addCallManager	ⓘ	✓	✓	✓	✓	✓	✓	✗	✓	✓
	getCallManager	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓
	listCallManager	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallManager	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateCallManager	ℹ	✓	✓	✓	✓	✓	✓	✓	✓	✓
CallManagerGroup	addCallManagerGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	getCallManagerGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	listCallManagerGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallManagerGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateCallManagerGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CallPark	addCallPark	ℹ	✓	✓	✓	✓	✓	ℹ	✓	✓	✓
	getCallPark	✓	✓	✓	✓	✓	✓	ℹ	✓	✓	✓
	listCallPark	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallPark	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateCallPark	✓	✓	✓	✓	✓	✓	ℹ	✓	✓	✓
CallPickupGroup	addCallPickupGroup	✓	✓	✓	ℹ	✓	✓	✓	✓	✓	✓
	getCallPickupGroup	✓	✓	✓	ℹ	✓	✓	✓	✓	✓	✓
	listCallPickupGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCallPickupGroup	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateCallPickupGroup	✓	✓	✓	ℹ	✓	✓	✓	✓	✓	✓
CcdAdvertisingService	addCcdAdvertisingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCcdAdvertisingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCcdAdvertisingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCcdAdvertisingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCcdAdvertisingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CcdHostedDN	addCcdHostedDN	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCcdHostedDN	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCcdHostedDN	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCcdHostedDN	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateCcdHostedDN	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CcdHostedDNGroup	addCcdHostedDNGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCcdHostedDNGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCcdHostedDNGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCcdHostedDNGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCcdHostedDNGroup	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CcdRequestingService	addCcdRequestingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCcdRequestingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCcdRequestingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCcdRequestingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCcdRequestingService	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CCMVersion	getCCMVersion	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
CiscoCatalyst600024PortFXSGateway	addCiscoCatalyst600024PortFXSGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCiscoCatalyst600024PortFXSGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCiscoCatalyst600024PortFXSGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCiscoCatalyst600024PortFXSGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCiscoCatalyst600024PortFXSGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CiscoCatalyst6000E1VoIPGateway	addCiscoCatalyst6000E1VoIPGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCiscoCatalyst6000E1VoIPGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCiscoCatalyst6000E1VoIPGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCiscoCatalyst6000E1VoIPGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCiscoCatalyst6000E1VoIPGateway	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
CiscoCatalyst6000T1VoIPGatewayPri	addCiscoCatalyst6000T1VoIPGatewayPri	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCiscoCatalyst6000T1VoIPGatewayPri	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCiscoCatalyst6000T1VoIPGatewayPri	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCiscoCatalyst6000T1VoIPGatewayPri	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCiscoCatalyst6000T1VoIPGatewayPri	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CiscoCatalyst6000T1VoIPGatewayT1	addCiscoCatalyst6000T1VoIPGatewayT1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	getCiscoCatalyst6000T1VoIPGatewayT1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	listCiscoCatalyst6000T1VoIPGatewayT1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCiscoCatalyst6000T1VoIPGatewayT1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	updateCiscoCatalyst6000T1VoIPGatewayT1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
CMCInfo (CmcInfo in Unified CM 8.0)	addCMCInfo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	getCMCInfo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	listCmcInfo	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCMCInfo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateCMCInfo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CommonDeviceConfig	addCommonDeviceConfig	✗	✗	✗	✓	✓	ℹ	ℹ	✓	✓	✓
	getCommonDeviceConfig	✗	✗	✗	✓	✓	ℹ	ℹ	✓	✓	✓
	listCommonDeviceConfig	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCommonDeviceConfig	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateCommonDeviceConfig	✗	✗	✗	✓	✓	ℹ	ℹ	✓	✓	✓
CommonPhoneConfig	addCommonPhoneConfig	✗	✗	✗	✗	✗	✗	✓	✓	ℹ	✓
	getCommonPhoneConfig	✗	✗	✗	✗	✗	✗	✓	✓	ℹ	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	listCommonPhoneConfig	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	removeCommonPhoneConfig	✗	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓
	updateCommonPhoneConfig	✗	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓
ConferenceBridge	addConferenceBridge	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓	✓	✓
	getConferenceBridge	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓	✓	✓
	listConferenceBridge	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeConferenceBridge	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateConferenceBridge	ⓘ	✓	✓	ⓘ	✓	ⓘ	✓	✓	✓	✓
createAutogeneratedProfile	createAutogeneratedProfile	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CredentialPolicy	addCredentialPolicy	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	getCredentialPolicy	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	listCredentialPolicy	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCredentialPolicy	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateCredentialPolicy	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
CSS (Css in Unified CM 8.0)	addCSS	✓	✓	✓	ⓘ	✓	✓	✓		✓	✓
	getCSS	✓	✓	✓	ⓘ	✓	✓	✓		✓	✓
	listCss	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
	removeCSS	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateCSS	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
CSSByName	listCSSByName	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓
CTIRoutePoint (CtiRoutePoint in Unified CM 8.0)	addCTIRoutePoint	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getCTIRoutePoint	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeCTIRoutePoint	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateCTIRoutePoint	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
DDI (Ddi in Unified CM 8.0(1))	addDDI	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	getDDI	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	removeDDI	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	updateDDI	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
DeviceByNameAndClass	listDeviceByNameAndClass	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗
DeviceLogin	doDeviceLogin	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DeviceLogout	doDeviceLogout	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓
DeviceMobility	addDeviceMobility	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getDeviceMobility	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeDeviceMobility	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateDeviceMobility	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
DeviceMobilityGroup	addDeviceMobilityGroup	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getDeviceMobilityGroup	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeDeviceMobilityGroup	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateDeviceMobilityGroup	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
DevicePool	addDevicePool	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	✓	✓
	getDevicePool	✓	✓	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	✓	✓
	removeDevicePool	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateDevicePool	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	ⓘ	✓	✓
DevicePoolByName	listDevicePoolByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DeviceProfile	addDeviceProfile	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	ⓘ	✓
	getDeviceProfile	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	ⓘ	✓
	listDeviceProfile	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	ⓘ	✓
	removeDeviceProfile	✓	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓
	updateDeviceProfile	ⓘ	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	ⓘ	✓
DeviceReset	doDeviceReset	ⓘ	✓	✓	✓	✓	✓	ⓘ	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
DHCPServer (DhcpServer in Unified CM 8.0(1))	addDHCPServer	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getDHCPServer	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	listDHCPServer	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeDHCPServer	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateDHCPServer	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
DHCPDHCPSubnet (DhcpSubnet in Unified CM 8.0(1))	addDHCPSubnet	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getDHCPSubnet	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	listDHCPSubnet	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeDHCPSubnet	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateDHCPSubnet	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
DialPlan	addDialPlan	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	getDialPlan	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeDialPlan	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	updateDialPlan	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
DialPlanTag	addDialPlanTag	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	getDialPlanTag	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeDialPlanTag	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
	updateDialPlanTag	✗	✗	✗	✗	✗	✗	✗	ⓘ	✓	✓
DirectedCallPark	addDirectedCallPark	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getDirectedCallPark	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeDirectedCallPark	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateDirectedCallPark	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
EnterpriseFeatureAccessCofiguration	addEnterpriseFeatureAccessCofiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	getEnterpriseFeatureAccessCofiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	listEnterpriseFeatureAccessCofiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	removeEnterpriseFeatureAccessCofiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateEnterpriseFeature AccessCofiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
FACInfo	addFACInfo	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getFACInfo	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeFACInfo	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateFACInfo	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
FallbackFeatureConfig	getFallbackFeatureConfig	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	updateFallbackFeatureConfig	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
Gatekeeper	addGatekeeper	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getGatekeeper	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeGatekeeper	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateGatekeeper	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
GatekeeperByName	listGatekeeperByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
GatewayEndpoint (use specific GatewayEndpoint API in Unified CM 8.0(1))	addGatewayEndpoint	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	✗	✓	✓
	getGatewayEndpoint	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	✗	✓	✓
	removeGatewayEndpoint	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓
	updateGatewayEndpoint	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	✗	✓	✓
GeoLocation	addGeoLocation	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	getGeoLocation	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	removeGeoLocation	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	updateGeoLocation	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
GeoLocationAttributeRule	addGeoLocationAttributeRule	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
	getGeoLocationAttributeRule	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
	removeGeoLocationAttributeRule	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateGeoLocationAttributeRule	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
GeoLocationFilter	addGeoLocationFilter	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	getGeoLocationFilter	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	removeGeoLocationFilter	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	updateGeoLocationFilter	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
GeoLocationPolicy	addGeoLocationPolicy	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	getGeoLocationPolicy	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	removeGeoLocationPolicy	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
	updateGeoLocationPolicy	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓	✓
GeoLocationPolicyMatrix	addGeoLocationPolicyMatrix	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
	getGeoLocationPolicyMatrix	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
	removeGeoLocationPolicyMatrix	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
	updateGeoLocationPolicyMatrix	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓
H323Gateway	addH323Gateway	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	getH323Gateway	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	removeH323Gateway	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateH323Gateway	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
H323Phone	addH323Phone	ⓘ	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	ⓘ	✓	✓
	getH323Phone	✓	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	ⓘ	✓	✓
	removeH323Phone	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateH323Phone	✓	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	ⓘ	✓	✓
H323Trunk	addH323Trunk	✓	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
	getH323Trunk	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
	listH323Trunk	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	removeH323Trunk	✓	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓
	updateH323Trunk	✓	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
HandOffConfiguration	addHandOffConfiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	getHandOffConfiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	removeHandOffConfiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	updateHandOffConfiguration	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
HuntList	addHuntList	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getHuntList	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeHuntList	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateHuntList	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
HuntPilot	addHuntPilot	✓	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	getHuntPilot	✓	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	removeHuntPilot	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateHuntPilot	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
ImeClient	addImeClient	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	getImeClient	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	listImeClient	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	removeImeClient	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
	updateImeClient	✗	✗	✗	✗	✗	✗	✗	✓	ⓘ	✓
ImeLearnedRoutes	getImeLearnedRoutes	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	removeImeLearnedRoutes	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	updateImeLearnedRoutes	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
IVRUserLocale (IvrUserLocale in Unified CM 8.0(1))	addIVRUserLocale	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getIVRUserLocale	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeIVRUserLocale	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateIVRUserLocale	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
LdapSync	doLdapSync	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
LdapSyncStatus	getLdapSyncStatus	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
LicenseCapabilities	getLicenseCapabilities	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateLicenseCapabilities	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
Line	addLine	✓	✓	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	✓	✓
	getLine	✓	✓	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	✓	✓
	removeLine	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateLine	ⓘ	✓	✓	ⓘ	ⓘ	✓	ⓘ	ⓘ	✓	✓
LineGroup	addLineGroup	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getLineGroup	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeLineGroup	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateLineGroup	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
Location	addLocation	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getLocation	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeLocation	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateLocation	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
LocationByName	listLocationByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MediaResourceList	addMediaResourceList	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getMediaResourceList	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeMediaResourceList	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateMediaResourceList	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
MediaResourceByName	listMediaResourceListByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MediaResourceGroup	addMediaResourceGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getMediaResourceGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeMediaResourceGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateMediaResourceGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
MediaResourceGroupByName	listMediaResourceGroupByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MeetMe	addMeetMe	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getMeetMe	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeMeetMe	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateMeetMe	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
MGCP	addMGCP	✓	✓	✓	ⓘ	✓	✓	ⓘ	✓	✓	✓
	getMGCP	✓	✓	✓	ⓘ	✓	✓	ⓘ	✓	✓	✓
	removeMGCP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateMGCP	ⓘ	✓	✓	✓	✓	✓	ⓘ	✓	✓	✓
MGCPEndpoint	addMGCPEndpoint	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	✓	✓	✓
	removeMGCPEndpoint	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MGCPSubunit	addMGCPSubunit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	removeMGCPSubunit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MGCPUnit	addMGCPUnit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	removeMGCPUnit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MobileSmartClientProfile	getMobileSmartClientProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
MobileVoiceAccess	addMobileVoiceAccess	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getMobileVoiceAccess	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeMobileVoiceAccess	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateMobileVoiceAccess	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
Mobility	addMobility	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	getMobility	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	removeMobility	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateMobility	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
MobilityProfile	addMobilityProfile	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	getMobilityProfile	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	listMobilityProfile	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	removeMobilityProfile	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
	updateMobilityProfile	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
MOHAudioSource	getMOHAudioSource	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	removeMOHAudioSource	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updateMOHAudioSource	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOHAudioSourceByName	listMOHAudioSourceByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOHServer	addMOHServer	✗	✗	✗	✓	✓	ℹ	✓	ℹ	✓	✓
	getMOHServer	✗	✗	✗	✓	✓	ℹ	✓	ℹ	✓	✓
	removeMOHServer	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
	updateMOHServer	✗	✗	✗	✓	✓	ℹ	✓	ℹ	✓	✓
NumDevices	getNumDevices	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Phone	addPhone	ℹ	✓	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	✓
	getPhone	ℹ	✓	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	✓
	listPhone	ℹ	✓	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	✓
	removePhone	✓	✓	✓	✓	✓	✓	✓	ℹ	ℹ	✓
	updatePhone	ℹ	✓	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	ℹ	✓
PhoneByDescription	listPhoneByDescription	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PhoneByName	listPhoneByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PhoneTemplate	addPhoneTemplate	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
	getPhoneTemplate	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
	removePhoneTemplate	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
	updatePhoneTemplate	✗	✗	✗	✓	✓	✓	✓	ℹ	✓	✓
PhoneTemplateByName	listPhoneTemplateByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
PhysicalLocation	addPhysicalLocation	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getPhysicalLocation	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removePhysicalLocation	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updatePhysicalLocation	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
PilotPoint	addPilotPoint	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓
	getPilotPoint	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓
	removePilotPoint	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	updatePilotPoint	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProcessNode	addProcessNode	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓	✓
	getProcessNode	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓	✓
	removeProcessNode	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateProcessNode	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓	✓
ProcessNodesByService	listProcessNodesByService	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProcessNodeService	getProcessNodeService	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateProcessNodeService	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
RecordingProfile	addRecordingProfile	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getRecordingProfile	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeRecordingProfile	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateRecordingProfile	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
Region	addRegion	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
	getRegion	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
	removeRegion	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateRegion	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
RegionMatrix	updateRegionMatrix	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
RemoteDestination	addRemoteDestination	✗	✗	✗	✓	✓	ⓘ	✓	✓	ⓘ	✓
	getRemoteDestination	✗	✗	✗	✓	✓	ⓘ	✓	✓	ⓘ	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	listRemoteDestination	✗	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓
	removeRemoteDestination	✗	✗	✗	✓	✓	✓	✓	✓	ⓘ	✓
	updateRemoteDestination	✗	✗	✗	✓	✓	ⓘ	✓	✓	ⓘ	✓
RemoteDestinationProfile	addRemoteDestinationProfile	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getRemoteDestinationProfile	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeRemoteDestinationProfile	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateRemoteDestinationProfile	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
ResourcePriorityDefaultNamespace	getResourcePriorityDefaultNamespace	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
	updateResourcePriorityDefaultNamespace	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
ResourcePriorityNamespace	addResourcePriorityNamespace	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	getResourcePriorityNamespace	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	removeResourcePriorityNamespace	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	updateResourcePriorityNamespace	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
ResourcePriorityNamespaceList	addResourcePriorityNamespaceList	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	getResourcePriorityNamespaceList	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	removeResourcePriorityNamespaceList	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	updateResourcePriorityNamespaceList	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
RouteList	addRouteList	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	ⓘ	✓
	getRouteList	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	ⓘ	✓
	listRouteList	✓	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓
	removeRouteList	✓	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓
	updateRouteList	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	ⓘ	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
RouteFilter	addRouteFilter	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getRouteFilter	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeRouteFilter	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateRouteFilter	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
RouteGroup	addRouteGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getRouteGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeRouteGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateRouteGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
RoutePartition	addRoutePartition	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
	getRoutePartition	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
	removeRoutePartition	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateRoutePartition	✓	✓	✓	ⓘ	✓	✓	✓	ⓘ	✓	✓
RoutePartitionByName	listRoutePartitionByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RoutePattern	addRoutePattern	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	getRoutePattern	✓	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
	removeRoutePattern	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateRoutePattern	ⓘ	✓	✓	✓	✓	ⓘ	ⓘ	ⓘ	✓	✓
RoutePlanByType	listRoutePlanByType	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ServiceParameter	getServiceParameter	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateServiceParameter	ⓘ	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
ServiceParameters	listServiceParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SIPProfile (SipProfile in Unified CM 8.0(1) onwards)	addSIPProfile	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	ⓘ	✓
	getSIPProfile	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	ⓘ	✓
	listSIPProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	ⓘ	✓
	removeSIPProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	ⓘ	✓
	updateSIPProfile	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	ⓘ	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
SIPRealm (SipRealm in Unified CM 8.0(1))	addSIPRealm	✗	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getSIPRealm	✗	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeSIPRealm	✗	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateSIPRealm	✗	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
SIPTrunk (SipTrunk in Unified CM 8.0(1))	addSIPTrunk	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
	getSIPTrunk	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
	listSIPTrunk	✓	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓
	removeSIPTrunk	✓	✓	✓	✓	✓	✓	✓	ⓘ	ⓘ	✓
	updateSIPTrunk	ⓘ	✓	✓	ⓘ	✓	ⓘ	ⓘ	ⓘ	ⓘ	✓
SIPTrunkSecurityProfile (SipTrunkSecurityProfile from Unified CM 8.0(1) onwards)	addSIPTrunkSecurityProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	getSIPTrunkSecurityProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	removeSIPTrunkSecurityProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	updateSIPTrunkSecurityProfile	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
SoftKeySet (SoftkeySet from Unified CM 8.5(1) onwards)	getSoftKeySet	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateSoftKeySet	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
SoftKeyTemplate	addSoftKeyTemplate	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	getSoftKeyTemplate	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	removeSoftKeyTemplate	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateSoftKeyTemplate	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
SQL	executeSQLUpdate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SQLQuery	executeSQLQuery	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TimePeriod	addTimePeriod	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getTimePeriod	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeTimePeriod	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
	updateTimePeriod	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
TimeSchedule	addTimeSchedule	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getTimeSchedule	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeTimeSchedule	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateTimeSchedule	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
TODAccess (TodAccess in Unified CM 8.0(1))	addTODAccess	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	getTODAccess	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	removeTODAccess	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	updateTODAccess	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
Transcoder	addTranscoder	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getTranscoder	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeTranscoder	✗	✗	✗	✓	✓	✓	✓	ⓘ	✓	✓
	updateTranscoder	✗	✗	✗	✓	✓	ⓘ	✓	ⓘ	✓	✓
TransformationPattern	addTransformationPattern	✗	✗	✗	✓	✓	ⓘ	✓	✓	✓	✓
	getTransformationPattern	✗	✗	✗	✓	✓	ⓘ	✓	✓	✓	✓
	removeTransformationPattern	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓
	updateTransformationPattern	✗	✗	✗	✓	✓	ⓘ	✓	✓	✓	✓
TransPattern	addTransPattern	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	getTransPattern	✓	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
	removeTransPattern	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateTransPattern	ⓘ	✓	✓	✓	✓	ⓘ	✓	ⓘ	✓	✓
UpdateRemoteCluster	doUpdateRemoteCluster	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
User	addUser	ⓘ	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	✓
	getUser	ⓘ	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	✓
	removeUser	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateUser	ⓘ	✓	✓	ⓘ	ⓘ	ⓘ	✓	ⓘ	✓	✓
UserByName	listUserByName	ⓘ	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 3-1 Operations by Release (continued)

APIs	Operations	5.0 d	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	8.0(1)	8.5(1)	8.6(1)
UserGroup	addUserGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getUserGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeUserGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateUserGroup	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
VG224 (Vg224 in Unified CM 8.0(1))	addVG224	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	✓	✓
	getVG224	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	✓	✓
	removeVG224	✗	✗	✗	✗	✗	✓	✓	ⓘ	✓	✓
	updateVG224	✗	✗	✗	✗	✗	✓	ⓘ	ⓘ	✓	✓
VoiceMailPilot	addVoiceMailPilot	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getVoiceMailPilot	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeVoiceMailPilot	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateVoiceMailPilot	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
VoiceMailPort	addVoiceMailPort	✓	✓	✓	ⓘ	✓	ⓘ	✓	ⓘ	✓	✓
	getVoiceMailPort	✓	✓	✓	ⓘ	✓	ⓘ	✓	ⓘ	✓	✓
	removeVoiceMailPort	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateVoiceMailPort	✓	✓	✓	ⓘ	✓	ⓘ	✓	ⓘ	✓	✓
VoiceMailProfile	addVoiceMailProfile	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	getVoiceMailProfile	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	removeVoiceMailProfile	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
	updateVoiceMailProfile	✓	✓	✓	✓	✓	✓	✓	ⓘ	✓	✓
VoiceMailProfileByName	listVoiceMailProfileByName	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

PART 2

Serviceability XML

CHAPTER 4

Serviceability XML Programming

This chapter describes the implementation of Serviceability XML APIs. Unified CM Real-Time information, Performance Counters, and Database information exposure occurs through an Serviceability XML interface that conforms to the World Wide Web Consortium (W3C) standard. The Web Service Description Language (WSDL) files provide interface definitions.

To access all AXL Configuration API downloads and AXL requests and responses that are found in this document, refer to http://developer.cisco.com. You must have a Cisco.com account and password to access this URL.

This chapter contains the following sections:

Overview

By exposing Unified CM (Unified CM) real-time information, performance counter, and database information, customers can write customized applications. Serviceability XML APIs, extensible SOAP-based XML Web Services, conform to the Simple Object Access Protocol (SOAP) Specification 1.1 and the Web Services Description Language (WSDL) Specification 1.1. Serviceability XML APIs represent one server component of the Unified CM Serviceability product.

SOAP provides an XML-based communication protocol and encoding format for interapplication communication. SOAP can serve as the backbone to a new generation of cross-platform, cross-language distributed-computing applications, termed Web Services.

Serviceability XML APIs provide remote procedure call (RPC) style operations for clients. Clients of Serviceability XML APIs can run in different OS platforms and can communicate through the standard SOAP protocol. Serviceability XML APIs provide access to core Unified CM Serviceability functionality through an open and standard transport protocol and data model.

The Serviceability XML interface uses the AXIS 1.4 SOAP server with the AXIS-2250 patch.

New and Changed Information

The following sections provide information on the changes in the Serviceability APIs in Unified CM release 8.6(1) and the previous releases:

- New Information for Unified CM 8.6(1), page 4-2
- New and Changed Information in Previous Releases of Unified CM, page 4-2

For information about the new, changed, or deprecated Serviceability SOAP API methods from the interface library, see the Chapter 5, “Serviceability XML Operations by Release”.

New Information for Unified CM 8.6(1)

The following change was made to the Serviceability XML API in Unified CM release 8.6(1).

- SelectCmDevice API returns device information for a maximum of 1000 devices instead of the earlier 200 devices.
  For more information, see the section MaxReturnedDevices, page 4-31.

New and Changed Information in Previous Releases of Unified CM

The following sections provide the new and changed information in the older releases of Unified CM:

- New Information forUnified CM 8.5(1), page 4-3
- New Information for Unified CM 8.0(1), page 4-5
- New Information for Unified CM 7.1(2), page 4-9
- New Information for Unified CM 7.0(1), page 4-11
- New Information for Unified CM 7.0, page 4-12
- New Information for Unified CM 6.1, page 4-13
- New Information for Unified CM 6.0, page 4-13

- New Information for Unified CM 5.0, page 4-14
- New Information for Unified CM 4.3, page 4-14
- New Information for Unified CM 4.0, page 4-14

For information about the new, changed, or deprecated Serviceability SOAP API methods from the interface library, see the Chapter 5, "Serviceability XML Operations by Release".

New Information forUnified CM 8.5(1)

The following are the Serviceability XML API changes in Unified CM release 8.5(1):

- The SelectCmDevice API returns SIP trunk status and also returns the IP addresses and status of the peer devices.
  For more information, see the sections SIPStatus, page 4-43 and SIPRemoteIpAddress, page 4-43.
- The SelectCmDevice API returns a new reason code when the phone in Power Save Plus mode unregisters from Unified CM.
  For more information, see the section StatusReason, page 4-39.

The following are the changes in Cisco Unified Serviceability in Unified CM release 8.5(1):

- The following new alarms are added to support the Single Sign-On and the Smart Card Authentication features:
  - LDAPServerUnreachable
  - SSODisabled
  - SSONullTicket
  - SSOServerUnreachable
  - SSOuserNotInDB
- The following existing alarms are modified to support the Single Sign-On and the Smart Card Authentication features:
  - authFail
  - authSuccess
  - authLdapInactive
- The following new alarms are added to support the OPTIONS Ping feature:
  - SIPTrunkISV
  - SIPTrunkOOS
  - SIPTrunkPartiallyISV
- The following new alarms are added to support the SIP Normalization and Transparency feature:
  - SIPNormalizationAutoResetDisabled
  - SIPNormalizationResourceWarning
  - SIPNormalizationScriptClosed
  - SIPNormalizationScriptError
  - SIPNormalizationScriptOpened
- New perfmon counters added to support the SIP Normalization and Transparency feature.
- New calledPartyPatternUsage CDR field added for the hunt list support.

- The following reason codes are added for the EndPointTransientConnection alarms:
  - AddressingModeMismatch
  - AutoRegisterDBConfigTimeout
  - autoRegisterDBError
  - CallManagerReset
  - CapabilityResponseTimeout
  - DatabaseTimeout
  - DBAccessError
  - DeviceInitiatedReset
  - DeviceTypeMismatch
  - DirectoryNumberMismatch
  - InvalidCapabilities
  - maxDevRegExceeded
  - RegistrationSequenceError
  - SecurityMismatch
- The following reason codes are added for the EndPointUnregistered alarms:
  - CallManagerForcedRestart
  - DatabaseConfigurationError
  - DeviceNameUnresolveable
  - InitializationError
  - MaxDevRegExceeded
  - NoEntryInDatabase
  - PowerSavePlus
  - SourceIPAddrChanged
  - SourcePortChanged
  - RegistrationSequenceError
  - InvalidCapabilities.
  - FallbackInitiated
  - DeviceSwitch
- Textual conventions updated for the following MIB objects:
  - CcmDevRegFailCauseCode
  - CcmDevUnregCauseCode

For more information on all the modifications in serviceability features, see the following documents:

- *New and Changed Information for Cisco Unified Communications Manager 8.5(1)*

Cisco Unified Serviceability Administration Guide, Release 8.5(1)

New Information for Unified CM 8.0(1)

The following are the updates in the Unified CM 8.0(1):

- The following new alarms are added:

Alarm Name	Severity
VAPTLSConnectionFailed	ALERT
VAPOverQuota	ALERT
CMVersionMismatch	ALERT
VAPInvalidCredentials	ALERT
VAPTCPSetupFailed	CRITICAL
InsufficientFallbackIdentifiers	ERROR
SAFInvalideMethodReceived	ERROR
VAPDHTInactive	ERROR
DbInfoCorrupt	ERROR
DbInfoTimeout	ERROR
DbInfoError	ERROR
SAFPublishRevoke	ERROR
EndPointTransientConnection	ERROR
ConflictingDataIE	ERROR
EndPointUnregistered	ERROR
ViPRQualityProblem	ERROR
ErrorParsingDirectiveFromPDP	ERROR
VAPPublishFailedOverQuota	ERROR
ConnectionFailureToPDP	ERROR
CCDRequestingServiceReceivedDuplicationPatterns	ERROR
CCDPSTNFailOverDurationTimeOut	ERROR
FailureResponseFromPDP	ERROR
SAFUnknownSubscription	ERROR
SAFUnknownService	ERROR
CCDAgeOutDurationTimeOut	ERROR
SafFwdError	ERROR
ConnectionToSaffBroke	ERROR
SAFBadFilter	ERROR
SAFResponderError	ERROR
InvalidVAPSubscription	WARNING
DigitAnalysisTimeoutAwaitingResponse	WARNING
FailedToFulfillDirectiveFromPDP	WARNING
VAPRejectedRoutes	WARNING

Alarm Name	Severity
ErrorParsingResponseFromPDP	WARNING
VAPPublishFailed	WARNING
CallAttemptBlockedByPolicy	WARNING
DbInsertValidatedDIDFailure	NOTICE
EndPointRegistered	NOTICE
CCDLearnedPatternExceededLimits	INFORMATIONAL
VAPPublicationRunCompleted	INFORMATIONAL
EndPointRestartInitiated	INFORMATIONAL
EndPointResetInitiated	INFORMATIONAL
RouteRemoved	INFORMATIONAL
ConnectionToPDPInService	INFORMATIONAL

- The following alarms are modified:

Alarm Name	Severity
NoFeatureLicense	EMERGENCY
DBLException	ALERT
CUCMOverallInitTimeExceeded	ALERT
SDLLinkOOS	ALERT
BChannelOOS	CRITICAL
CodeYellowEntry	CRITICAL
MaxCallsReached	CRITICAL
DChannelOOS	CRITICAL
CallManagerFailure	CRITICAL
StationTCPInitError	CRITICAL
CodeRedEntry	CRITICAL
MGCPGatewayLostComm	CRITICAL
TimerThreadSlowed	CRITICAL
NumDevRegExceeded	ERROR
ICTCallThrottlingStart	ERROR
DeviceTypeMismatch	ERROR
DeviceCloseMaxEventsExceeded	ERROR
ConnectionFailure	ERROR
DeviceInitTimeout	ERROR
MultipleSIPTrunksToSamePeerAndLocalPort	ERROR
InvalidIPNetPattern	ERROR
H323Stopped	WARNING
StationEventAlert	WARNING

Alarm Name	Severity
DevicePartiallyRegistered	WARNING
MaliciousCall	WARNING
UserUserPrecedenceAlarm	WARNING
MediaResourceListExhausted	WARNING
NotEnoughChans	WARNING
SIPStopped	WARNING
DaTimeOut	WARNING
RouteListExhausted	WARNING
DeviceTransientConnection	WARNING
SIPLineRegistrationError	WARNING
DeviceUnregistered	WARNING
BeginThrottlingCallListBLFSubscriptions	WARNING
AnnunciatorNoMoreResourcesAvailable	WARNING
ConferenceNoMoreResourcesAvailable	WARNING
MtpNoMoreResourcesAvailable	WARNING
SIPStarted	NOTICE
SDLLinkISV	NOTICE
H323Started	NOTICE
ICTCallThrottlingEnd	NOTICE
CodeYellowExit	NOTICE
MaxCallDurationTimeout	NOTICE
BChannelISV	NOTICE
DChannelISV	NOTICE
CallManagerOnline	NOTICE
MGCPGatewayGainedComm	NOTICE
DeviceRegistered	INFORMATIONAL
EndThrottlingCallListBLFSubscriptions	INFORMATIONAL
DeviceApplyConfigInitiated	INFORMATIONAL
CUCMTotalInitializationStateTime	INFORMATIONAL
MaxHoldDurationTimeout	INFORMATIONAL
PktCapServiceStarted	INFORMATIONAL
PktCapOnDeviceStarted	INFORMATIONAL
StationConnectionError	INFORMATIONAL
DatabaseDefaultsRead	INFORMATIONAL
PktCapServiceStopped	INFORMATIONAL
DeviceDnInformation	INFORMATIONAL
StationAlarm	INFORMATIONAL

Alarm Name	Severity
DeviceResetInitiated	INFORMATIONAL
PktCapOnDeviceStopped	INFORMATIONAL
DeviceRestartInitiated	INFORMATIONAL
CUCMInitializationStateTime	INFORMATIONAL

- The following new perform classes are added:
  - CCM_VIPR_SERVICE_OBJECT
  - EXTERNALCALLCONTROL_OBJ
  - CCM_SAFCLIENT_OBJECT
  - ViPR
- The following perform classes are modified:
  - CCM_PRESENCE_OBJECT
  - CCM_MGCPFXO_OBJECT
  - CCM_DUAL_MODE_MOBILITY_OBJECT
  - CCM_VCB_OBJECT
  - CCM_PHONE_OBJECT
  - CcmSIPObject
  - CCM_LOCATIONS_OBJECT
  - CCM_WSM_CONNECTOR_OBJECT
  - CcmQSIGFeatureObject
  - CCM_SIP_STATION_OBJECT
  - UCB_SW_DEVICE_OBJECT
  - MGCP_GATEWAY_OBJECT
  - MOH_DEVICE_OBJECT
  - CCM_LINES_OBJECT
  - CCM_MGCPBRI_OBJECT
  - CCM_MGCPT1CAS_OBJECT
  - CCM_HUNTLISTS_OBJECT
  - CcmAnnunciatorObject
  - CcmSIPStackObject
  - MTP_DEVICE_OBJECT
  - CCM_SDL_OBJECT
  - CCM_SIGNALING_OBJECT
  - CCM_ANALOGACCESS_OBJECT
  - UCB_HW_DEVICE_OBJECT
  - CCM_H323_OBJECT

  - CISCO_CALL_RESTRICTION
  - CCM_MGCPFXS_OBJECT
  - CCM_MGCPPRI_OBJECT
  - GATEKEEPER_OBJECT
  - CCM_MOBILITY_OBJECT
  - CCM_CM_OBJECT
  - XCODE_DEVICE_OBJECT
  - PRFOBJ_EMAPP
- The following changes are made in the CISCO-CCM-MIB:
  - ccmDevFailCauseCode is supported in Unified CM 8.0(1), but will be deprecated in future release.
  - Two new textual conventions are added for Device Registration status that replaces ccmDevFailCauseCode:

    ccmDevRegFailCauseCode—Identifies the reasons for a device registration failure.

    ccmDevUnregCauseCode—Identifies the reasons a device was unregistered.
  - ccmGatewayFailed trap is supported in Unified CM 8.0(1), but will be deprecated in future release.
  - New ccmGatewayFailedReason trap is added to replace ccmGatewayFailed.
- The following enhancements are made in Cisco Syslog-MIB:
  - Cisco Syslog Agent sends the end point alarms to Remote Syslog Server and RTMT
  - Alarm severity or alarm string configurations done at RTMT are used by Cisco Syslog Agent to compare against the end point alarm severity and string. If they match, the end point alarm is sent to remote syslog server and RTMT.
  - Cisco Syslog agent uses the Exclude End Point alarms parameter to determine if end point alarms are sent to the Remote Syslog server and RTMT.
- The following changes are made in Serviceability API Cipher Support:

  Unified CM 8.0(1) Web server (Tomcat) supports the following Ciphers

  Most Secure:
  - TLS_RSA_WITH_AES_256_CBC_SHA
  - TLS_DHE_DSS_WITH_AES_256_CBC_SHA

  More Secure:
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_DHE_RSA_WITH_AES_128_CBC_SHA

  Secure:
  - SSL_RSA_WITH_3DES_EDE_CBC_SHA
  - SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA

New Information for Unified CM 7.1(2)

The following information applies to Unified CM 7.1(2):

- Added new service URL that contains the version of Unified CM.
- Added a new API, SelectCmDevice, to show IPv6 details of Unified CM node or server, phone devices, SIP devices, and media devices.
- Added a new API, SelectCtiDevice, to support IPv6 address search for CTI controlled application and devices.
- The following perfmon counters have been added to show the IPv6 network statistics:
  - In Receives
  - In HdrErrors
  - In UnknownProtos
  - In Discards
  - In Delivers
  - Out Requests
  - Out Discards
  - Reasm Reads
  - Reasm OKs
  - Reasm Fails
  - Frag OKs
  - Frag Fails
  - Frag Creates
  - InOut Requests
- Introduced new phone seamless upgrade information:
  - Phone Active Load ID
  - Phone InActive Load ID
  - Phone Down Load Status
  - Phone Down Load Failure Reason
  - Phone Firmware Down Loaded Server
- CISCO-CCM-MIB has been enhanced to show IPv6 details of Unified CM node, phone devices, SIP devices, and media devices.

The SOAP service APIs supported by Unified CM Release 7.1(2) is listed in Table 4-1.

Table 4-1 Serviceability XML Methods supported by Cisco Unified Communications Manager Release 7.1(2)

SOAP Service	API	Service URL with Version
RisPort (Real Time Information Port)	selectCmDevice (IPv4 devices)	https://<server>:8443/realtimeservice/services/RisPort
	SelectCmDevice (includes IPv6 devices)	https://<server>:8443/realtimeservice/services/RisPort70
	selectCtiItem (IPv4 devices)	https://<server>:8443/realtimeservice/services/RisPort
	SelectCtiItem (includes IPv6 devices)	https://<server>:8443/realtimeservice/services/RisPort70
	getServerInfo	https://<server>:8443/realtimeservice/services/RisPort

Table 4-1 Serviceability XML Methods supported by Cisco Unified Communications Manager Release 7.1(2)

SOAP Service	API	Service URL with Version
PerfmonPort (Performance Information Port)	perfmonOpenSession	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonAddCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonRemoveCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCollectSessionData	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCloseSession	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonListInstance	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonQueryCounterDescription	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonListCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCollectCounterData	https://<server>:8443/perfmonservice/services/PerfmonPort
ControlCenterServices Port (All Service Control APIs)	soapGetStaticServiceList	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapGetServiceStatus	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapDoServiceDeployment	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapDoControlServices	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	getProductInformationList	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
LogCollectionPort (All Log Collection APIs)	listNodeServiceLogs	https://<server>:8443/logcollectionservice/services/LogCollectionPort
	selectLogFiles	https://<server>:8443/logcollectionservice/services/LogCollectionPort
CDRonDemand (All CDR APIs)	get_file_list	https://<server>:8443/CDRonDemandService/services/CDRonDemand
	get_file	https://<server>:8443/CDRonDemandService/services/CDRonDemand
DimeGetFileService (Geting Single File)	GetOneFile	https://<server>:8443/logcollectionservice/services/DimeGetFileService

New Information for Unified CM 7.0(1)

The following information applies to Unified CM Release 7.0(1):

- The following disk partition counters have been obsolesced starting with Unified CM, release 7.0(1):
  - Await Read Time—Average time measured in milliseconds, for read requests issued to the device to be served. The counter is no longer valid with a counter value as –1.
  - Await Write Time—Average time measured in milliseconds, for write requests issued to the device to be served. The counter is no longer valid with a counter value as –1.

  - Await Time—Average time measured in milliseconds, for input/output (I/O) requests issued to the device to be served. Includes time spent by the requests in queue and the time spent servicing them. The counter is no longer valid with a counter value as –1.
  - % CPU Time—Percentage of CPU time that is dedicated to handling IO requests that were issued to the disk. The counter is no longer valid with a counter value as –1.
  - Queue Length—Average queue length for the requests that were issued to the disk. The counter is no longer valid with a counter value as –1.
- The following counters have been added to memory perfmon object:
  - Pages Input Per Sec—Total number of kilobytes paged in from disk per second.
  - Pages Output Per Sec—Total number of kilobytes paged out to disk per second.
  - Faults Per Sec—Number of page faults (major + minor) made by the system per second (post 2.5 kernels only).

    This is not the number of page faults that generate I/O, because some page faults can be resolved without I/O.
  - Major Faults Per Sec—Number of major faults the system has made per second, those which have required loading a memory page from disk (post 2.5 kernels only).
  - Low Total—Total low (non-paged) memory for kernel.
  - Low Free—Free low (non-paged) memory for kernel.
- Added Cisco SOAPMessage tracing service (SOAP Service Tracing, page 4-142).

New Information for Unified CM 7.0

The following information applies to Unified CM 7.0:

- There are no Serviceability XML API changes in this release.
- When trace compression is enabled, trace files are compressed.
- Cisco Unified CM 7.0 supports version in the service URL as described in Table 4-2.

Note The service URL for all the Serviceability XML Unified CM 4.x and below is http://ASTSERVERNAME/soap/astsvc.dll.

Table 4-2 Serviceability XML Methods by Cisco Unified Communications Manager Release

SOAP Service	API	Service URL with Version
RisPort (Real Time Information Port)	selectCmDevice	https://<server>:8443/realtimeservice/services/RisPort
	selectCtiItem	https://<server>:8443/realtimeservice/services/RisPort
	getServerInfo	https://<server>:8443/realtimeservice/services/RisPort

Table 4-2 Serviceability XML Methods by Cisco Unified Communications Manager Release (continued)

SOAP Service	API	Service URL with Version
PerfmonPort (Performance Information Port)	perfmonOpenSession	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonAddCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonRemoveCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCollectSessionData	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCloseSession	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonListInstance	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonQueryCounterDescription	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonListCounter	https://<server>:8443/perfmonservice/services/PerfmonPort
	perfmonCollectCounterData	https://<server>:8443/perfmonservice/services/PerfmonPort
ControlCenterServices Port (All Service Control APIs)	soapGetStaticServiceList	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapGetServiceStatus	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapDoServiceDeployment	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	soapDoControlServices	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
	getProductInformationList	https://<server>:8443/controlcenterservice/services/ControlCenterServicesPort
LogCollectionPort (All Log Collection APIs)	listNodeServiceLogs	https://<server>:8443/logcollectionservice/services/LogCollectionPort
	selectLogFiles	https://<server>:8443/logcollectionservice/services/LogCollectionPort
CDRonDemand (All CDR APIs)	get_file_list	https://<server>:8443/CDRonDemandService/services/CDRonDemand
	get_file	https://<server>:8443/CDRonDemandService/services/CDRonDemand
DimeGetFileService (Geting Single File)	GetOneFile	https://<server>:8443/logcollectionservice/services/DimeGetFileService

New Information for Unified CM 6.1

There are no Serviceability XML API changes for Unified CM 6.1.

New Information for Unified CM 6.0

The following updates apply to Unified CM 6.0:

- The GetProductInformationList API has been added. This API provides information about the products that are installed on a given server.
- The standard SOAP fault is issued if a failure occurs. For related information, see the "SOAP Fault Error Codes" section on page 4-144.

New Information for Unified CM 5.0

The following APIS have been added for Unified CM 5.0:

- getServerInfo—Exports information from the Server Information SOAP interface
- soapGetStaticServiceList—Allows clients to perform a query for all services static specifications in Unified CM.
- soapGetServiceStatus—Allows clients to perform a list of deployable and undeployable services status query
- soapDoServiceDeployment—Allows clients to deploy or undeploy a list of services
- soapDoControlServices—Allows clients to start or stop a list of service
- listNodeServiceLogs—Returns the location of their log files for each service.
- selectLogFiles—Takes FileSelectionCriteria object as an input parameter and returns the file names and location for that object.
- get_file_list—Allows an application to query the CDR Repository Node for a list of all the files that match a specified time interval
- GetOneFile—Takes as an input parameter the absolute file name of the file that you want to collect from the server

New Information for Unified CM 4.3

There are no Serviceability XML API changes for Unified CM 4.3.

New Information for Unified CM 4.0

The following APIs have been added for Unified CM 4.0:

- selectCmDevice—Allows clients to perform Unified CM device-related queries
- selectCtiItem—Allows clients to perform a CTI manager-related query
- SelectCmDeviceSIP—Allows clients to perform Unified CM SIP device related queries

Data Model

Serviceability XML APIs are based on SOAP with XML request and response messages. APIs must conform to the structure of a SOAP Envelope element. Although SOAP is a standard protocol, many of its data model aspects remain open for flexibility reasons. For example, it permits different transport protocols, such as SMTP, FTP, or HTTP, to carry SOAP messages. The implementation of a SOAP service must specify the bindings of the data model to constitute a concrete wire protocol specification.

The Web Services Description Language (WSDL) Specification 1.1 provides the mechanism to describe the complete SOAP bindings that Serviceability XML APIs use.

SOAP Binding

The binding section of the Serviceability SOAP WSDL files specifies Serviceability XML API SOAP binding information. Binding specifications apply to both request and response messages. SOAP Binding covers the aspects that are explained in the following sections.

Character Encoding

Serviceability XML APIs use UTF-8 to encode the data stream in both request and response SOAP messages. The encoding attribute of the XML declaration specifies UTF-8 encoding. Serviceability XML APIs also set “text/xml; charset='utf-8'” as the value of the Content-Type response header field. Internally, Serviceability XML APIs processes the data by using UCS-2 Unicode code page.

Binding Style

Serviceability XML APIs uses remote procedure call (RPC) binding style. In SOAP, the word operation refers to method or function. Each Serviceability XML API operation call gets modeled as an RPC that is encapsulated in SOAP messages. The HTTP request carries RPC calls while the HTTP response carries the call returns. The call information is modeled as a structure. The member elements of the body entry represent the accessor elements that represent the input parameters. The response data is also modeled as a structure with accessors that correspond to output parameters. Parameters that are both input and output must appear in both the request and response message.

Transport Protocols

SOAP allows different transport protocols to carry SOAP messages. Serviceability XML APIs use the standard HTTP as its transport. Clients use the POST verb to send requests to Serviceability XML APIs.

Note Serviceability XML APIs do not use the M-POST method as defined in the HTTP Extension Framework.

Encoding Rule

Serviceability XML APIs follow the recommended data model and serialization/encoding rules as defined in Section 5.1 of SOAP Specification 1.1 for both the request and response messages. SOAP simple types are based on the built-in data types that are defined in XML Schema Part 2.

Serviceability XML APIs define their own data types, which are derived from the built-in types. The schemas element of the Serviceability XML APIs WSDL file specifies Serviceability XML APIs that are derived data types. Serviceability XML APIs use both simple and compound data types, such as arrays and structures. All operations in Serviceability XML APIs pass parameters by value.

For performance reasons, Serviceability XML APIs do not specify the size of the array elements in the response message. It leaves the size as [], which means that no particular size is specified, but the clients can determine the size by enumerating the actual number of elements that are inside the array. Point 8 of Section 5.1 of SOAP Specification 1.1 specifies this.

The target namespace URL for Serviceability XML API data types schema follows:

http://schemas.xmlsoap.org/soap/envelope/

Serviceability XML APIs qualify the first body entry in the response, which represents the call-return, with this target namespace. Similarly, clients of Serviceability XML APIs also need to qualify the first body entry in the request, which represents the call, with this namespace.

Request Message

With RPC-style SOAP binding, the request message contains operation call information that is encoded as a struct. The call struct, which appears as the first body entry in the request message, contains the name of the operation and the input parameters. The name of the top-level element represents the operation name. The struct contains accessor element members that represent input parameters. Operations with no parameters have no members in the struct. The names of the accessor elements match the names of the input parameters. The values of the accessor elements represent the values of the input parameters. The order of the accessor elements must match the order of the input parameters as specified in the signature of the operation.

SOAP Action Header

Serviceability XML APIs require SOAP clients to include the SOAP Action HTTP header field in the request message. The SOAP 1.1 specification does not place any restrictions on the format of this header. For Serviceability XML APIs, the **soapAction** attribute of the SOAP element, which is defined under the binding section of the Serviceability SOAP API WSDL files, specifies the format of the SOAP Action HTTP header. All Serviceability XML APIs operations use the following URI format:

http://schemas.cisco.com/ast/soap/action/#PortName#OperationName

Note Because the enclosing double-quote characters (“ “) are part of the URI, the header must include them.

PortName acts as a placeholder that refers to the name of the port. Serviceability XML APIs must support the port. OperationName represents a placeholder that refers to the name of the intended operation within the specified port. This name must match the operation name in the request body, which is specified as the name of the first body entry element. The SOAP Action header indicates the intent of the SOAP request without having to parse the body of the request. A SOAP server can check the value of this header and determine whether to fail the operation before it proceeds with parsing the XML body. This provides some efficiency on the server side and allows non-SOAP-aware intermediary HTTP servers or proxies to determine the intent of the payload.

Port

A SoapPort basically represents an instantiation of a SoapBinding with a specific network address. Because Serviceability XML APIs use HTTP as the transport protocol, the network address in this case specifies an HTTP request URL. SoapPortType, with specific binding rules added to it, provides a basis for the definition of SoapBinding.

The service section of the WSDL file defines the request URL for all Serviceability XML API operations. Every request for the Serviceability XML APIs operations must use this URL.

SOAP Header

As previously explained, the SOAP header provides a general way to add features to SOAP messages in a decentralized fashion with no prior contract between the sender and recipient. Serviceability XML APIs do not use this feature, so no Header element is expected in the envelope and gets ignored. If the Header element gets included with the **mustUnderstand** attribute set to 1, Serviceability XML APIs reply with a fault and a fault code value that is set to the **mustUnderstand** value.

The following example shows an Serviceability XML API SOAP request message with the HTTP header information:

Example

```
POST  /perfmonservice/services/PerfmonPort HTTP/1.1
Charset: utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
user-agent: ClientName
Host: nozomi
Content-Type: text/xml; charset=utf-8
Content-Length: xxx
SOAPAction: "http://schemas.cisco.com/ast/soap/action/#PerfmonPort#PerfmonOpenSession"

<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
            <q1:PerfmonOpenSession xmlns:q1="http://schemas.cisco.com/ast/soap/" />
        </soap:Body>
</soap:Envelope>
```

Response Message

For a successful operation call, the call-return gets encoded as a structure. The structure appears as the first body entry of the response. The call-return basically contains the output parameters or return values of the call. The name of the structure top-level element has no significance, and the SOAP 1.1 specification does not place any restriction on the name. The structure contains accessor member elements, which represent the output parameters of the call. The names of the accessor elements match the names of the output parameters. The contents of the accessor elements represent the values of the output parameters. The order of the accessor elements must match the order of output parameters as specified in the operation signatures. Operation, which does not return any value, contains no member elements in the call-return struct.

Serviceability XML APIs use the following naming conventions for the call-return top-level element:

```
SOAPAction: "http://schemas.cisco.com/ast/soap/action/#PortName #OperationName"
```

where `OperationName` represents a placeholder that refers to the name of the operation that is specified in the request message. This format specifically applies only for the Serviceability XML APIs implementation.

The target namespace that is described in the "Encoding Rule" section on page 4-15, qualifies the call-return. Serviceability XML APIs return HTTP status 200 when the operation succeeds.

For a failed operation call, Serviceability XML APIs include the SOAP fault element in the response body. Similar to call-return, the fault element also appears as the first body entry. Serviceability XML APIs set HTTP 500 status when sending fault messages.

Fault Message

When an Serviceability XML API processes a request and detects that an error occurred, it replies with a SOAP fault element in the response. The fault element appears as the first response body entry. The fault element comprises the following four subelements:

- Fault Code Values
- FaultString
- FaultActor
- Detail

Fault Code Values

Serviceability XML APIs use the following standard fault code values as defined in Section 4.4.1 of the SOAP 1.1 specification.

VersionMismatch

This fault code gets set when the namespace URL of the request envelope does not match.

MustUnderstand

This fault code gets set when the clients include the `Header` element in the envelope along with the mustUnderstand attribute set to 1. Serviceability XML APIs do not use the Header element. See the "SOAP Header" section on page 4-17 for details.

Client

This fault code gets set when Serviceability XML APIs encounters errors that are related to the clients.

Server

This fault code gets set when Serviceability XML APIs encounter errors that are related to the service itself. This subelement always exists in the fault element as specified in the SOAP 1.1 specification. This represents a general classification of errors.

FaultString

Serviceability XML APIs set a short error description in the faultstring element that is intended for human reading. Similar to faultcode, this subelement is always present as the SOAP 1.1 specification requires. The string value of the FaultString specifically applies only to the Serviceability XML APIs.

FaultActor

Serviceability XML APIs do not set this subelement. Note that a proxy or intermediary server must include this subelement if it generates a fault message.

Detail

If an Serviceability XML API parses and processes the request body and determines that an error occurs afterward, it includes the detailed error information in the detail subelement.

If Serviceability XML APIs do not include the detail subelement in the fault element, the error does not relate to the request body. Data types that are defined in the Serviceability XML APIs WSDL files specify the encoding rule for the detail subelement.

The following fragments of the file describe the data types:

```
...
<complexType name='CallInfoType'>
    <sequence>
       <element name='FileName' type='xsd:string'/>
       <element name='LineNo' type='xsd:int'/>
       <element name='ErrorCode' type='xsd:unsignedInt'/>
       <element name='Function' type='xsd:string'/>
       <element name='Params' type='xsd:string'/>
    </sequence>
</complexType>

<complexType name='ErrorInfoType'>
    <sequence>
       <element name='Version' type='xsd:string'/>
       <element name='Time' type='xsd:time'/>
       <element name='ProcId' type='xsd:unsignedInt'/>
       <element name='ThreadId' type='xsd:unsignedInt'/>
       <element name='ArrayOfCallInfo' type='tns:ArrayOfCallInfoType'/>
    /sequence>
</complexType>
...
<complexType name='ArrayOfCallInfoType'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
            <sequence>
                <element name='CallInfo'
                    type='tns:CallInfoType' minOccurs='1' maxOccurs='unbounded'/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
```

Serviceability XML APIs name the detail entry element as ErrorInfo and the type as ErrorInfoType. This type provides a structure with several accessor elements. The Version accessor contains the build version. The Time accessor denotes the time when the error occurs. The ProcId accessor contains the process ID of the Serviceability XML APIs. The ThreadId accessor contains the thread ID that generates the fault. The ArrayOfCallInfo accessor contains an array of CallInfo elements.

The type for CallInfo specifies CallInfoType and also represents a structure. CallInfoType contains detailed information that describes where the error occurs in the code. It also includes the returned error code of the function, and the parameter data. The CallInfoType design allows capturing as much information as needed, so it is easy and fast to track down and investigate a problem. Depending on the implementation of the operation, several CallInfo elements can exist in the array.

The following example shows a successful Serviceability XML API SOAP response message with HTTP headers:

Example

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <m:PerfmonOpenSessionResponse xmlns:m="http://schemas.cisco.com/ast/soap/">
            <SessionHandle>{01944B7E-183F-44C5-977A-F31E3AE59C4C}</SessionHandle>
        </m:PerfmonOpenSessionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The following example shows a failed Serviceability XML API SOAP response message with HTTP headers:

Example

```
HTTP/1.1 500 OK
Content-Type: text/xml; charset=utf-8
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
            <faultcode>SOAP-ENV:Server</faultcode>
            <faultstring>Perfmon error occurs</faultstring>
            <detail>
                <m:ErrorInfo xmlns:m="http://schemas.cisco.com/ast/soap/">
                    <Version>3.2.0.2</Version>
                    <Time>07/16/2001 - 00:00:24</Time>
                    <ProcId>1200</ProcId>
                    <ThreadId>300</ThreadId>
                    <ArrayOfCallInfo SOAP-ENC:arrayType="m:CallInfoType[]">
                        <CallInfo>
                            <FileName>perfmon.cpp</FileName>
                            <LineNo>396</LineNo>
                            <ErrorCode>3221228473</ErrorCode>
                            <Function>AddCounter</Function>
                            <Params>\\nozomi\tcp\Bad Counter Name</Params>
                        </CallInfo>
                    </ArrayOfCallInfo>
                </m:ErrorInfo>
            </detail>
        </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Namespaces

Serviceability XML APIs use the following XML namespaces:

- `http://schemas.xmlsoap.org/soap/envelope/`
  The namespace URI for the SOAP envelope
- `http://schemas.xmlsoap.org/soap/encoding/`
  The namespace for the SOAP-recommended encoding rule that is based on XML Schema
- `http://schemas.cisco.com/ast/soap/`
  The namespace URL for Serviceability XML API data types as defined in the WSDL file

Downloading Serviceability SOAP WSDL Files

You can download the Unified CM serviceability SOAP WSDL files from the Unified CM server directly by entering a URL on the web browser. Table 4-3 lists these URLs. In each URL, *servername* must be replaced by an appropriate server IP address.

Table 4-3 URLs for SOAP Requests Service Definitions

SOAP Request Type	URL for SOAP Requests Service Definition
RisPort RisPort70	https://*servername*:8443/realtimeservice/services/RisPort?wsdl https://*servername:*8443/realtimeservice/services/RisPort70?wsdl
PerfmonPort	https://*servername*:8443/perfmonservice/services/PerfmonPort?wsd
ControlCenterServices	https://*servername*:8443/controlcenterservice/services/ControlCenterServicesPort?wsdl
LogCollectionService	https://*servername*:8443/logcollectionservice/services/LogCollectionPort?wsdl
CDRonDemand	https://*servername*:8443/CDRonDemandService/services/CDRonDemand?wsdl
DimeGetFile	https://*servername*:8443/logcollectionservice/services/DimeGetFileService?wsdl
SOAPMonitorService[1]	https://*servername*:8443/realtimeservice/services/SOAPMonitorService?wsdl

1. SOAPMonitorService is the standard SOAP monitor service WSDL of the third-party Axis SOAP server.

PClients of Serviceability XML APIs must download these files to know what services are available, how to form the request message, and how to interpret the response message properly. Basically, the WSDL file has what you need to know about Serviceability XML APIs.

Monitoring SOAP Activity

You can use AXIS SOAPMonitor to monitor SOAP activities. Point your browser to

https://*servername*:8443/realtimeservice/SOAPMonitor

where *servername* is an appropriate server IP address.

RisPort SOAP Service

The RisPort (Real-Time Information Port) service comprises several operations that allow clients to do the following tasks related to real-time information:

Table 4-4 provides a summary of the SOAP RisPort service operations.

Table 4-4 SOAP RisPort Service Operations

Operation	Description	Reference
selectCmDevice	Allows clients to perform Unified CM device-related queries	RisPort Service: selectCmDevice Operation, page 4-23
SelectCmDevice (includes IPv6 devices)	Allows clients to search for devices that have IPv6 addresses	RisPort Service: SelectCmDevice Operation (Includes IPv6 Devices), page 4-29
selectCtiItem	Allows clients to perform a CTI manager-related query	RisPort Service: selectCtiItem Operation, page 4-48
SelectCtiItem (includes IPv6 devices)	Supports search for CTI controlled applications and devices that have IPv6 addresses	RisPort Service: SelectCtiDevice Operation (Includes IPv6 Devices), page 4-50
getServerInfo	Exports information from the Server Information SOAP interface	RisPort Service: getServerInfo Operation, page 4-60
SelectCmDeviceSIP	Allows clients to perform Unified CM SIP device related queries	RisPort Service: SelectCmDeviceSIP Operation, page 4-64

RisPort Service: selectCmDevice Operation

The selectCMDevice operation allows clients to perform Unified CM device-related queries.

Request Format

Note For information about obtaining all device information in a large system, refer to the "Device Query Support for Large Clusters" section on page 4-151

SOAP Action and Envelope Information

HTTP header should have following SOAP action for these queries.

```
SOAPAction: "http://schemas.cisco.com/ast/soap/action/#RisPort#SelectCmDevice"
```

Query information includes an Envelope as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://schemas.cisco.com/ast/soap/"
xmlns:types="http://schemas.cisco.com/ast/soap/encodedTypes"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

Session ID

This SOAP header will have session ID that is a unique session ID from client.

```
<soap:Header>
<tns:AstHeader id="id1">
<SessionId xsi:type="xsd:string">SessionId</SessionId>
</tns:AstHeader>
</soap:Header>
```

Selection Criteria

Selection criteria type, which is a Unified CM Selection criteria, follows the SOAP header. Selection criteria should not include "unknown." If you specify "unknown," it is treated as "any." The "Unknown" state can be present in a response.

```
<soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<tns:SelectCmDevice>
```

If the same information is queried over again, send Stateinfo from the previous request for each repetitive query by client.

```
<StateInfo xsi:type="xsd:string" />
<CmSelectionCriteria href="#id1"/>
</tns:SelectCmDevice><tns:CmSelectionCriteria id="id1" xsi:type="tns:CmSelectionCriteria">
```

Maximum Devices Specification

Specifies the maximum number of devices for which information can be returned in a search. In Unified CM release 8.5(1) the maximum number is 1000. Prior to Unified CM release 8.5(1) the maximum number was 200.

```
<MaxReturnedDevices xsi:type="xsd:unsignedInt">10</MaxReturnedDevices>
```

Search Device Classes

This example specifies the device class type to query for real-time status. Device classes include 'Any', 'Phone', 'Gateway', 'H323', 'Cti', 'VoiceMail', 'MediaResources', 'HuntList', 'SIPTrunk', and 'unknown'.

```
<Class xsi:type="tns:DeviceClass">Any</Class>
```

This example specifies the Model of the device—255 specifies all models.

```
<Model xsi:type="xsd:unsignedInt">255</Model>
```

Device Status in Search Criteria

Specify registered/unregistered status devices. The following example shows status 'Any', 'Registered', 'UnRegistered', 'Rejected', and 'Unknown.'

```
<Status xsi:type="tns:CmDevRegStat">Registered</Status>
```

The following example specifies the server name where the search needs to be performed. If no name is specified, a search in all servers in a cluster results.

```
<NodeName xsi:type="xsd:string" />
```

Specify Selection type whether it is IP Address/Name/SIPStatus for SIP Trunk

```
    <element name="SelectBy" type="tns:CmSelectBy"/>
    <simpleType name="CmSelectBy">
    <restriction base="string">
    <enumeration value="Name"/>
    <enumeration value="IPV4Address"/>
    <enumeration value="DirNumber"/>
    <enumeration value="Description"/>
    <enumeration value="SIPStatus"/>
    </restriction>
    </simpleType>
```

Array of Items for Which Search Criteria Are Specified

The following example specifies an array that contains the IP Address or Device Name of the items for which a real-time status is required.

```
<SelectItems href="#id2" />Name or IP</tns:CmSelectionCriteria
<soapenc:Array id="id2" soapenc:arrayType="tns:SelectItem[2]">
<Item href="#id3"/><Item xsi:null="1"/>
</soapenc:Array>
<tns:SelectItem id="id3" xsi:type="tns:SelectItem">
<Item xsi:type="xsd:string"/></tns:SelectItem>
</soap:Body>
</soap:Envelope>
```

Response Format

The Response follows the following schema and contains information for one to many servers for each server and contains a sequence of search information that is found on the search criteria.

```
<complexType name='SelectCmDeviceResult'>
    <sequence>
           <element name='TotalDevicesFound' type='xsd:unsignedInt'/>
           <element name='CmNodes' type='tns:CmNodes'/>
  </sequence>
</complexType>
```

CMNodes provides a list of Unified CMNodes that are given in search criteria.

```
<complexType name='CmNodes'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
         <sequence>
       <element name='CmNode' type='tns:CmNode' minOccurs='0 maxOccurs='1000'/>
         </sequence>
        </restriction>
```

```
    </complexContent>
 </complexType>
```

Each Unified CMNode contains a sequence of devices and their registration status.

```
<complexType name='CmNode'>
    <sequence>
        <element name='ReturnCode'          type='tns:RisReturnCode'/>
        <element name='Name'                type='xsd:string'/>
        <element name='NoChange'            type='xsd:boolean'/>
        <element name='CmDevices'           type='tns:CmDevices'/>
   </sequence>
</complexType>
<complexType name='CmDevices'>
    <complexContent>
      <restriction base='SOAP-ENC:Array'>
      <sequence>
      <element name='CmDevice' type='tns:CmDevice' minOccurs='0' maxOccurs='200'/>
      </sequence>
      </restriction>
      </complexContent>
    </complexType>
```

The Unified CM Device information contains the following information.

```
<complexType name='CmDevice'>
<sequence>
<element name='Name' type='xsd:string'/>
<element name='IpAddress' type='xsd:string'/>
<element name='DirNumber' type='xsd:string'/>
<element name='Class' type='tns:DeviceClass'/>
<element name='Model' type='xsd:unsignedInt'/>
<element name='Product' type='xsd:unsignedInt'/>
<element name='BoxProduct'type='xsd:unsignedInt'/>
<element name='Httpd' type='tns:CmDevHttpd'/>
<element name='RegistrationAttempts' type='xsd:unsignedInt'/>
<element name='IsCtiControllable'  type='xsd:boolean'/>
<element name='LoginUserId' type='xsd:string'/>
<element name='Status'  type='tns:CmDevRegStat'/>
<element name='StatusReason' type='xsd:unsignedInt'/>
<element name='SIPStatus' type='tns:SIPStatus'/>
<element name='SIPRemoteIpAddress' type='xsd:string'/>
<element name='PerfMonObject' type='xsd:unsignedInt'/>
<element name='DChannel' type='xsd:unsignedInt'/>
<element name='Description' type='xsd:string'/>
<element name='H323Trunk' type='tns:H323Trunk'/>
<element name='TimeStamp' type='xsd:unsignedInt'/>
</sequence>
</complexType>
```

Example Request

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:SelectCmDevice soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <StateInfo xsi:type="xsd:string"/>
      <CmSelectionCriteria href="#id0"/>
    </ns1:SelectCmDevice>
```

```
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:CmSelectionCriteria"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://schemas.cisco.com/ast/soap/">
      <MaxReturnedDevices xsi:type="xsd:unsignedInt">200</MaxReturnedDevices>
      <Class xsi:type="xsd:string">Any</Class>
      <Model xsi:type="xsd:unsignedInt">255</Model>
      <Status xsi:type="xsd:string">Registered</Status>
      <NodeName xsi:type="xsd:string" xsi:nil="true"/>
      <SelectBy xsi:type="xsd:string">Name</SelectBy>
      <SelectItems soapenc:arrayType="ns2:SelectItem[1]" xsi:type="soapenc:Array">
        <item href="#id1"/>
      </SelectItems>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:SelectItem" xmlns:ns3="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Item xsi:type="xsd:string">*</Item>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:SelectCmDeviceResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SelectCmDeviceResult xsi:type="ns1:SelectCmDeviceResult">
        <TotalDevicesFound xsi:type="xsd:unsignedInt">4</TotalDevicesFound>
        <CmNodes soapenc:arrayType="ns1:CmNode[1]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns1:CmNode">
            <ReturnCode xsi:type="ns1:RisReturnCode">Ok</ReturnCode>
            <Name xsi:type="xsd:string">CISCART15</Name>
            <NoChange xsi:type="xsd:boolean">false</NoChange>
            <CmDevices soapenc:arrayType="ns1:CmDevice[4]" xsi:type="soapenc:Array">
              <item xsi:type="ns1:CmDevice">
                <Name xsi:type="xsd:string">ANN_2</Name>
                <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
                <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
                <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
                <Model xsi:type="xsd:unsignedInt">126</Model>
                <Product xsi:type="xsd:unsignedInt">89</Product>
                <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
                <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
                <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
                <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
                <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
                <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
                <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
                <PerfMonObject xsi:type="xsd:unsignedInt">608</PerfMonObject>
                <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
                <Description xsi:type="xsd:string">ANN_CISCART15</Description>
                <H323Trunk xsi:type="ns1:H323Trunk">
                  <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
```

```
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
      <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
      <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
      <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
    </H323Trunk>
    <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
  </item>
  <item xsi:type="ns1:CmDevice">
    <Name xsi:type="xsd:string">CFB_2</Name>
    <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
    <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
    <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
    <Model xsi:type="xsd:unsignedInt">50</Model>
    <Product xsi:type="xsd:unsignedInt">28</Product>
    <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
    <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
    <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
    <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
    <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
    <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
    <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
    <PerfMonObject xsi:type="xsd:unsignedInt">15</PerfMonObject>
    <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
    <Description xsi:type="xsd:string">CFB_CISCART15</Description>
    <H323Trunk xsi:type="ns1:H323Trunk">
      <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
      <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
      <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
      <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
    </H323Trunk>
    <TimeStamp xsi:type="xsd:unsignedInt">1204679736</TimeStamp>
  </item>
  <item xsi:type="ns1:CmDevice">
    <Name xsi:type="xsd:string">MOH_2</Name>
    <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
    <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
    <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
    <Model xsi:type="xsd:unsignedInt">70</Model>
    <Product xsi:type="xsd:unsignedInt">51</Product>
    <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
    <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
    <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
    <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
    <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
    <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
    <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
    <PerfMonObject xsi:type="xsd:unsignedInt">6</PerfMonObject>
    <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
    <Description xsi:type="xsd:string">MOH_CISCART15</Description>
    <H323Trunk xsi:type="ns1:H323Trunk">
      <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
```

```
                    <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
                    <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
                    <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
                    <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
                    <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
                    <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
                    <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
                  </H323Trunk>
                  <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
                </item>
                <item xsi:type="ns1:CmDevice">
                  <Name xsi:type="xsd:string">MTP_2</Name>
                  <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
                  <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
                  <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
                  <Model xsi:type="xsd:unsignedInt">110</Model>
                  <Product xsi:type="xsd:unsignedInt">30</Product>
                  <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
                  <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
                  <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
                  <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
                  <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
                  <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
                  <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
                  <PerfMonObject xsi:type="xsd:unsignedInt">13</PerfMonObject>
                  <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
                  <Description xsi:type="xsd:string">MTP_CISCART15</Description>
                  <H323Trunk xsi:type="ns1:H323Trunk">
                    <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
                    <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
                    <Zone xsi:type="xsd:string" xsi:nil="true"/>
                    <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
                    <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
                    <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
                    <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
                    <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
                    <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
                    <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
                  </H323Trunk>
                   <SIPStatus xsi:type='tns:SIPStatus'>InService</SIPStatus>
                   <SIPRemoteIpAddress xsi:type='xsd:string'> 0.77.31.18=Available</SIPRemoteIpAddress>
                    <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
                  </item>
                </CmDevices>
              </item>
            </CmNodes>
          </SelectCmDeviceResult>
          <StateInfo xsi:type="xsd:string">&lt;StateInfo ClusterWide=&quot;1&quot;&gt;&lt;Node
Name=&quot;CISCART15&quot; SubsystemStartTime=&quot;1204679712&quot; StateId=&quot;4&quot;
TotalItemsFound=&quot;4&quot;
TotalItemsReturned=&quot;4&quot;/&gt;&lt;/StateInfo&gt;</StateInfo>
        </ns1:SelectCmDeviceResponse>
      </soapenv:Body>
    </soapenv:Envelope>
```

RisPort Service: SelectCmDevice Operation (Includes IPv6 Devices)

The **SelectCMDevice** API allows clients to query Cisco Unified Communications Manager (Unified CM) for device-related information. This API supports searching for IPv6 device address (Unified CM node/server, phone devices, SIP devices, and media devices) and provides IPv6 information in the response. This API also supports searching by download-status for new generation phones with seamless upgrade capability and provides information on the download-status of the firmware.

The operation name for invoking the API is **SelectCmDevice** and the service URL is:

https://<server name>/realtimeservice/services/RisPort70.

Note The service URL for release 7.1(2) is different from the earlier releases. The service URL of this release includes the version information.

The **SelectCmDevice** operation comprises the **SelectCmDeviceInput** and **SelectCmDeviceOutput** messages:

```
<message name="SelectCmDeviceInput">
    <part name="StateInfo" type="xsd:string"/>
    <part name="CmSelectionCriteria" type="tns:CmSelectionCriteria"/>
</message>
<message name="SelectCmDeviceOutput">
    <part name="SelectCmDeviceResult" type="tns:SelectCmDeviceResult"/>
    <part name="StateInfo" type="xsd:string"/>
</message>
<portType name="RisPortType">
    <operation name="SelectCmDevice">
        <input message="tns:SelectCmDeviceInput"/>
        <output message="tns:SelectCmDeviceOutput"/>
    </operation>
</portType>
```

This section has the following information:

Request Format

Note For information about obtaining all device information in a large system, refer to the "Device Query Support for Large Clusters" section on page 4-151

SOAP Action

HTTP header should have following SOAP action for these queries.

```
SOAPAction: http://schemas.cisco.com/ast/soap/action/#RisPort70#SelectCmDevice
```

Envelope and Session ID

Query information includes the envelope and session ID information as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
<soapenv:Header>
<AstHeader xsi:type="soap:AstHeader">
<SessionId xsi:type="xsd:string"/>
</AstHeader>
</soapenv:Header>
```

The SOAP header has the session ID that is a unique ID.

SelectCmDevice Operation

The **SelectDevice** operation is first defined in the SOAP body element.

```
<SOAP-ENV:Body soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<soap:SelectCmDevice>
```

Stateinfo

If same Information is queried repetitively, then **Stateinfo** is sent from the previous request. **Stateinfo** is the string that is returned by the server and it represents the state of the real-time database.

```
<StateInfo xsi:type="xsd:string" />
```

Selection Criteria

The **CmSelectionCriteria** element defines the selection criteria for Realtime Information Server Data Collection (RISDC) search. The selection criteria type follows the SOAP header. Selection criteria should not be “unknown.” If you specify “unknown,” it is treated as “any.” The “Unknown” state can be present in a response.

```
<complexType name="CmSelectionCriteria">
    <sequence>
        <element name="MaxReturnedDevices" nillable="true" type="xsd:unsignedInt"/>
        <element name="Class" nillable="true" type="xsd:string"/>
        <element name="Model" nillable="true" type="xsd:unsignedInt"/>
        <element name="Status" nillable="true" type="xsd:string"/>
        <element name="NodeName" nillable="true" type="xsd:string"/>
        <element name="SelectBy" type="tns:CmSelectBy"/>
        <element name="SelectItems" nillable="true" type="tns:SelectItems"/>
        <element name="Protocol" nillable="true" type="tns:ProtocolType"/>
        <element name="DownloadStatus" nillable="true" type="tns:DeviceDownloadStatus"/>
    </sequence>
</complexType
```

CmSelectionCriteria contains the following information:

- MaxReturnedDevices
- Class
- Model
- Status
- NodeName
- SelectBy
- SelectItems
- Protocol
- DownloadStatus

MaxReturnedDevices

Specifies the maximum number of devices for which information can be returned in a search. In Unified CM release 8.6(1) the maximum number is 1000. Prior to Unified CM release 8.6(1) the maximum number was 200.

Format:

```
<element name="MaxReturnedDevices" nillable="true" type="xsd:unsignedInt"/>
```

Example:

```
<MaxReturnedDevices xsi:type="xsd:unsignedInt">1000</MaxReturnedDevices>
```

Class

Specifies the device class type that needs to be queried for the real-time status. The following options are available:

- Any
- Phone
- Gateway
- H323
- Cti
- VoiceMail
- MediaResources
- SIP Trunk
- HuntLis
- Unknown

Format:

```
<element name="Class" nillable="true" type="xsd:string"/>
<simpleType name="DeviceClass">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Phone"/>
<enumeration value="Gateway"/>
<enumeration value="H323"/>
<enumeration value="Cti"/>
<enumeration value="VoiceMail"/>
<enumeration value="MediaResources"/>
<enumeration value="SIP Trunk"/>
<enumeration value="HuntList"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<Class xsi:type="tns:DeviceClass">Any</Class>
```

Model

Specifies the model of the device. 255 means all models.

Format:

```
<element name="Model" nillable="true" type="xsd:unsignedInt"/>
```

Example:

```
<Model xsi:type="xsd:unsignedInt">255</Model>
```

Status

Specifies the status of the device. The following options are available:

- Any
- Registered
- UnRegistered
- Rejected
- PartiallyRegistered
- Unknown

Format:

```
<element name="Status" nillable="true" type="xsd:string"/>
```

Example:

```
<Status xsi:type="tns:CmDevRegStat">Registered</Status>
```

NodeName

Specifies the server name where search is performed. If no name is specified, then all the servers within the cluster will be searched.

Format:

```
<element name="NodeName" nillable="true" type="xsd:string"/>
```

Example:

```
<NodeName xsi:type="xsd:string" xsi:nil="true"/>
```

SelectItems

Specifies the array of items for which you can specify the search criteria.

Format:

```
<complexType name="SelectItems">
<complexContent>
<restriction base="soapenc:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:SelectItem[]"/>
</restriction>
</complexContent>
</complexType>
```

Example that contains IP address or name of the device of the items for which you need the real time status:

```
<SelectItems href="#id2" />Name or IP</tns:CmSelectionCriteria
<soapenc:Array id="id2" soapenc:arrayType="tns:SelectItem[2]">
<Item href="#id3"/><Item xsi:null="1"/>
</soapenc:Array>
<tns:SelectItem id="id3" xsi:type="tns:SelectItem">
<Item xsi:type="xsd:string"/></tns:SelectItem>
```

SelectBy

Specifies the Unified CM selection types during the search to RISDC. The following options are available:

- Name
- IPV4Address
- IPV6Address
- DirNumber

- Description
- SIPStatus for SIP Trunk

Format:

```
<element name="SelectBy" type="tns:CmSelectBy"/>
<simpleType name="CmSelectBy">
<restriction base="string">
<enumeration value="Name"/>
<enumeration value="IPV4Address"/>
<enumeration value="IPV6Address"/>
<enumeration value="DirNumber"/>
<enumeration value="Description"/>
<enumeration value="SIPStatus"/>
</restriction>
</simpleType>
```

Example:

```
<SelectBy href="#id1"/>
</multiRef>
<multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:CmSelectBy"
xmlns:ns3="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">IPV6Address</multiRef>
```

Protocol

Specifies the protocol name in the search criteria. The following options are available:

- Any
- SCCP
- SIP
- Unknown

Format:

```
<element name="Protocol" type="tns:ProtocolType"/>
<simpleType name="ProtocolType">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="SCCP"/>
<enumeration value="SIP"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<Protocol href="#id3"/>
</multiRef>
<multiRef id="id3" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns5:ProtocolType"
xmlns:ns5="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Any</multiRef>
```

DownloadStatus

Specifies the download status of the application. The following options are available:

- Any
- Upgrading

- Successful
- Failed
- Unknown

Format:

```
<element name="DownloadStatus" nillable="true" type="tns:DeviceDownloadStatus"/>
<simpleType name="DeviceDownloadStatus">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Upgrading"/>
<enumeration value="Successful"/>
<enumeration value="Failed"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<DownloadStatus href="#id4"/>
</multiRef>
<multiRef id="id4" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns6:DeviceDownloadStatus"
xmlns:ns6="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Any</multiRef>
```

Response Format

The response has the following schema and contains information for a single server or many servers. The response contains a sequence of search information that is specified in the search criteria.

```
<complexType name='SelectCmDeviceResult'>
<sequence>
<element name='TotalDevicesFound' type='xsd:unsignedInt'/>
<element name='CmNodes' type='tns:CmNodes'/>
<element name='
</sequence>
</complexType>
```

Total Devices Found

This element displays the total number of device that are found.

Format:

```
<element name="TotalDevicesFound" type="xsd:unsignedInt"/>
```

Example:

```
<TotalDevicesFound xsi:type="xsd:unsignedInt">4</TotalDevicesFound>
```

Call Manager Node Information

CMNodes provides a list of Call Manager nodes that are specified in the search criteria.

Format:

```
<complexType name="CmNodes">
<complexContent>
<restriction base="SOAP-ENC:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:CmNode[]"/>
</restriction>
</complexContent>
</complexType>
```

Each CMNode contains a sequence of devices and their registration status. The CmNode element displays the following information:

- Return Code
- Name
- NoChange
- CmDevice

Format:

```
<complexType name='CmNode'>
<sequence>
<element name='ReturnCode' type='tns:RisReturnCode'/>
<element name='Name' type='xsd:string'/>
<element name='NoChange' type='xsd:boolean'/>
<element name='CmDevices' type='tns:CmDevices'/>
</sequence>
</complexType>
```

Return Code

Displays the RIS return codes. The following options are available:

- Ok
- NotFound
- InvalidRequest
- InternalRequest
- InternalError
- NodeNotResponding
- InvalidNodeName

Format:

```
<element name='ReturnCode' type='tns:RisReturnCode'/>
<simpleType name="RisReturnCode">
<restriction base="string">
<enumeration value="Ok"/>
<enumeration value="NotFound"/>
<enumeration value="InvalidRequest"/>
<enumeration value="InternalError"/>
<enumeration value="NodeNotResponding"/>
<enumeration value="InvalidNodeName"/>
</restriction>
```

Name

Displays the name of the node.

Example:

```
Name xsi:type="xsd:string">172.27.203.17</Name>
```

NoChange

Example:

```
<NoChange xsi:type="xsd:boolean">false</NoChange>
```

CmDevice

Displays the device information. See Call Manager Device Information.

Call Manager Device Information

The CmDevice displays the following information:

- Name
- Class
- Product
- Httpd
- IsCtiControllable
- Status
- PerfMonObject
- Description
- TimeStamp
- NumOfLines
- ActiveLoadID
- DownloadStatus
- DownloadServer
- SIPStatus
- DirNumber
- Model
- BoxProduct
- RegistrationAttempts
- LoginUserId
- StatusReason
- DChannel
- H323Trunk
- Protocol
- LinesStatus
- InactiveLoadID
- DownloadFailureReason
- IPAddress
- SIPRemoteIpAddress

Format:

```
<complexType name="CmDevice">
<sequence>
<element name="Name" type="xsd:string"/>
<element name="DirNumber" nillable="true" type="xsd:string"/>
<element name="Class" nillable="true" type="tns:DeviceClass"/>
<element name="Model" nillable="true" type="xsd:unsignedInt"/>
<element name="Product" nillable="true" type="xsd:unsignedInt"/>
<element name="BoxProduct" nillable="true" type="xsd:unsignedInt"/>
<element name="Httpd" nillable="true" type="tns:CmDevHttpd"/>
<element name="RegistrationAttempts" type="xsd:unsignedInt"/>
<element name="IsCtiControllable" type="xsd:boolean"/>
<element name="LoginUserId" nillable="true" type="xsd:string"/>
<element name="Status" type="tns:CmDevRegStat"/>
<element name="StatusReason" nillable="true" type="xsd:unsignedInt"/>
<element name="PerfMonObject" nillable="true" type="xsd:unsignedInt"/>
<element name="DChannel" nillable="true" type="xsd:unsignedInt"/>
<element name="Description" nillable="true" type="xsd:string"/>
<element name="H323Trunk" nillable="true" type="tns:H323Trunk"/>
<element name="TimeStamp" nillable="true" type="xsd:unsignedInt"/>
<element name="Protocol" type="tns:ProtocolType"/>
<element name="NumOfLines" type="xsd:unsignedInt"/>
<element name="LinesStatus" type="tns:CmDevLinesStatus"/>
<element name="ActiveLoadID" nillable="true" type="xsd:string"/>
<element name="InactiveLoadID" nillable="true" type="xsd:string"/>
```

```
<element name="DownloadStatus" nillable="true" type="tns:DeviceDownloadStatus"/>
<element name="DownloadFailureReason" nillable="true" type="xsd:string"/>
<element name="DownloadServer" nillable="true" type="xsd:string"/>
<element name="IPAddress" type="tns:IPAddressArray"/>
<element name='SIPStatus' type='tns:SIPStatus'/>
<element name='SIPRemoteIpAddress' type='xsd:string'/>
</sequence>
</complexType>
```

Name

Displays the name of the device.

Example:

```
<Name xsi:type="xsd:string">CTIRP</Name>
```

DirNumber

Displays the directory number.

Example:

```
<DirNumber xsi:type="xsd:string">9999</DirNumber>
```

Class

Defines the device class types. The following options are available:

- Any
- Phone
- Gateway
- H323
- Cti
- VoiceMail
- MediaResources
- SIP Trunk
- HuntLis
- Unknown

Format:

```
<element name="Class" nillable="true" type="xsd:string"/>
<simpleType name="DeviceClass">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Phone"/>
<enumeration value="Gateway"/>
<enumeration value="H323"/>
<enumeration value="Cti"/>
<enumeration value="VoiceMail"/>
<enumeration value="MediaResources"/>
<enumeration value="SIP Trunk"/>
<enumeration value="HuntList"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<Class xsi:type="ns1:DeviceClass">Cti</Class>
```

Model

Specifies the model of the device.

Example:

```
<Model xsi:type="xsd:unsignedInt">73</Model>
```

Product

Displays the product type.

Example:

```
<Product xsi:type="xsd:unsignedInt">48</Product>
```

BoxProduct

Displays the integer value (0 or 1) for box products.

```
<BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
```

Httpd

Displays the devices support HTTP. The following otpions are available:

- Yes
- No
- Unknown

Format:

```
<simpleType name="CmDevHttpd">
<restriction base="string">
<enumeration value="Yes"/>
<enumeration value="No"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
```

RegistrationAttempts

Displays the number of registration attempts.

Example:

```
<RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
```

IsCtiControllable

Displays either yes or no.

Example:

```
<IsCtiControllable xsi:type="xsd:boolean">true</IsCtiControllable>
```

LoginUserId

Displays the login user ID.

Format:

```
<LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
```

Status

Displays the device registration status type. The following options are available:

- Any
- Registered
- Unregistered
- Rejected
- PartiallyRegistered
- Unknown

Format:

```
<simpleType name="CmDevRegStat">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Registered"/>
<enumeration value="UnRegistered"/>
<enumeration value="Rejected"/>
<enumeration value="PartiallyRegistered"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

Example:

```
<Status xsi:type="ns1:CmDevRegStat">Registered</Status>
```

StatusReason

Displays the status of the device. It is an integer value. If the device is in “Registered” state then <StatusReason> is 0. If the device is in “Unregistered” state then one of the following values is displayed:

Value	Description
1	Unknown - The device has unregistered for an unknown reason. If the device does not re-register within 5 minutes, verify that it is powered up and verify that there is network connectivity between the device and Unified CM.
6	ConnectivityError - Network communication between the device and Unified CM has been interrupted. Possible causes include device power outage, network power outage, network configuration error, network delay, packet drops, and packet corruption. It is also possible to get this error if the Unified CM node is experiencing high CPU usage. Verify that the device is powered up and operating, verify that there is network connectivity between the device and Unified CM, and verify that the CPU utilization is in the safe range (you can monitor this via the CPU Pegging Alert in RTMT).
8	DeviceInitiatedReset - The device has initiated a reset, possibly due to a power cycle or internal error. No action required; the device will re-register automatically.
9	CallManagerReset - A device reset was initiated from Cisco Unified CM Administration, either due to an explicit command from an administrator, or due to internal errors encountered. No action necessary; the device will re-register automatically.
10	DeviceUnregistered - The device has explicitly unregistered. Possible causes include a change in the IP address or port of the device. No action is necessary; the device will re-register automatically.

Value	Description
11	MalformedRegisterMsg - (SIP only) A SIP REGISTER message could not be processed because of an illegal format. Possible causes include a missing Call-ID header, a missing AoR in the To header, or an expires value that is too small. Check the REGISTER message for any of these issues and if you find one, correct the issue.
12	SCCPDeviceThrottling - (SCCP only) The indicated SCCP device exceeded the maximum number of events allowed per-SCCP device. Events can be phone calls, KeepAlive messages, or excessive SCCP or non-SCCP messages. The maximum number of allowed events is controlled by the Unified CM service parameter, Max Events Allowed. When an individual device exceeds the number configured in that service parameter, Unified CM closes the TCP connection to the device; automatic reregistration generally follows. This action is an attempt to stop malicious attacks on Unified CM or to ward off excessive CPU usage.
13	KeepAliveTimeout - A KeepAlive message was not received. Possible causes include device power outage, network power outage, network configuration error, network delay, packet drops, and packet corruption. It is also possible to get this error if the Unified CM node is experiencing high CPU usage. Verify that the device is powered up and operating, verify that there is network connectivity between the device and Unified CM, and verify the CPU utilization is in the safe range (you can monitor this via the CPU Pegging Alert in RTMT).
14	ConfigurationMismatch (SIP only) The configuration on the device does not match the configuration in Unified CM. This can be caused by database replication errors or other internal Unified CM communication errors. Go to the Cisco Unified Reporting web page, generate a Unified CM Database Status report and verify "all servers have a good replication status". You can also go to the Real-Time Reporting Tool (RTMT) and check the Replication Status in the Database Summary page. If status shows 2, then replication is working. If this device continues to unregister with this reason code, go to the Device Configuration page in Unified CM Administration for the device indicated in this alarm and click Save. This generates a change notify message to the Unified CM and TFTP services and rebuilds a new configuration file for the device. If the problem still persists, restart the TFTP service and the Unified CM service.
15	CallManagerRestart - A device restart was initiated from Unified CM Administration, either due to an explicit command from an administrator or due to a configuration change such as adding, deleting or changing a directory number associated with the device. No action is necessary; the device will re-register automatically.
16	DuplicateRegistration - Unified CM detected that the device attempted to register to two nodes at the same time. Unified CM initiated a restart to the phone to force it to re-home to a single node. No action is necessary; the device will re-register automatically.
17	CallManagerApplyConfig - An ApplyConfig action was performed in Unified CM Administration resulting in an unregistration. No action is necessary; the device will re-register automatically.
18	DeviceNoResponse - The device did not respond to a reset or restart notification, so it is being forcefully reset. If the device does not re-register within 5 minutes, confirm that it is powered up and confirm that there is network connectivity between the device and Unified CM.
21	PowerSavePlus - The device powered off as a result of the Power Save Plus feature that is enabled for this device. When the device powers off, it remains unregistered from Unified CM for the duration specified in the Phone On Time parameter in the Product Specific Configuration for this device.

Example:

```
<StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
```

PerfMonObject

Displays the PerfMonObject ID.

```
<PerfMonObject xsi:type="xsd:unsignedInt">2</PerfMonObject>
```

DChannel

Displays the number of D channels supported for PRI devices.

Example:

```
<DChannel xsi:type="xsd:unsignedInt">0</DChannel>
```

Description

Example:

```
<Description xsi:type="xsd:string">CTIRP1</Description>
```

H323Trunk

Displays the H323 trunk details.

Format:

```
<complexType name="H323Trunk">
    <sequence>
        <element name="ConfigName" nillable="true" type="xsd:string"/>
        <element name="TechPrefix" nillable="true" type="xsd:string"/>
        <element name="Zone" nillable="true" type="xsd:string"/>
        <element name="RemoteCmServer1" nillable="true" type="xsd:string"/>
        <element name="RemoteCmServer2" nillable="true" type="xsd:string"/>
        <element name="RemoteCmServer3" nillable="true" type="xsd:string"/>
        <element name="AltGkList" nillable="true" type="xsd:string"/>
        <element name="ActiveGk" nillable="true" type="xsd:string"/>
        <element name="CallSignalAddr" nillable="true" type="xsd:string"/>
        <element name="RasAddr" nillable="true" type="xsd:string"/>
    </sequence>
</complexType>
```

TimeStamp

Displays the UTC format timestamp.

Example:

```
<TimeStamp xsi:type="xsd:unsignedInt">1222331666</TimeStamp>
```

Protocol

Displays the device protocol types. The following options are available:

- Any
- SCCP
- SIP
- Unknown

Example:

```
<Protocol xsi:type="ns1:ProtocolType">Any</Protocol>
```

NumOfLines

Displays the number of lines.

Example:

```
<NumOfLines xsi:type="xsd:unsignedInt">0</NumOfLines>
```

LinesStatus

Displays the single line status types for SIP devices. The following options are available:

- Any
- Registered
- UnRegistered
- Rejected
- Unknown

Format:

```
<simpleType name="CmSingleLineStatus">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Registered"/>
<enumeration value="UnRegistered"/>
<enumeration value="Rejected"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

ActiveLoadID

Displays the currently active firmware on the phone.

Format:

```
<element name="ActiveLoadID" nillable="true" type="xsd:string"/>
```

InactiveLoadID

Displays the inactive phone partition load of the phone (if phone supports dual partition).

Format:

```
<element name="InactiveLoadID" nillable="true" type="xsd:string"/>
```

DownloadStatus

Displays the inactive partition phone download status.

Format:

```
<element name="DownloadStatus" nillable="true" type="tns:DeviceDownloadStatus"/>
```

DownloadFailureReason

Displays the inactive partition download failure, if any.

Format:

```
<element name="DownloadFailureReason" nillable="true" type="xsd:string"/>
```

DownloadServer

Displays the inactive partition downloading server.

Format:

```
<element name="DownloadServer" nillable="true" type="xsd:string"/>
```

IPAddress

Displays the IP address type, the IP address, and the attribute type.

Format:

```
<complexType name="IPAddressArray">
    <complexContent>
        <restriction base="SOAP-ENC:Array">
            <attribute ref="soapenc:arrayType"
wsdl:arrayType="tns:IPAddressArrayType[]"/>
        </restriction>
    </complexContent>
</complexType>

<complexType name="IPAddressArrayType">
    <sequence>
        <element name="IP" type="xsd:string"/>
        <element name="IPAddrType" type="tns:IPAddrType"/>
        <element name="Attribute" type="tns:AttributeType"/>
    </sequence>
</complexType>
<simpleType name="AttributeType">
    <restriction base="string">
        <enumeration value="Unknown"/>
        <enumeration value="Adminstrative"/>
        <enumeration value="Signaling"/>
        <enumeration value="AdministrativeAndSignaling"/>
    </restriction>
</simpleType>
<simpleType name="IPAddrType">
    <restriction base="string">
        <enumeration value="ipv4"/>
        <enumeration value="ipv6"/>
    </restriction>
</simpleType>
```

SIPStatus

Displays the status of SIP trunk. The following options are available:

- InService
- OutOfService
- PartialService
- Unknown

Format:

```
<simpleType name="SIPStatus">
    <restriction base="string">
         <enumeration value="InService"/>
         <enumeration value="OutOfService"/>
         <enumeration value="PartialService"/>
         <enumeration value="Unknown"/>
    </restriction>
</simpleType>
```

SIPRemoteIpAddress

Displays a list of remote IP addresses and its status.

Format:

```
    <element name='SIPRemoteIpAddress' type='xsd:string'/>
```

Example Request

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:SelectCmDevice soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <StateInfo xsi:type="xsd:string"/>
      <CmSelectionCriteria href="#id0"/>
    </ns1:SelectCmDevice>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:CmSelectionCriteria"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://schemas.cisco.com/ast/soap/">
      <MaxReturnedDevices xsi:type="xsd:unsignedInt">1000</MaxReturnedDevices>
      <Class xsi:type="xsd:string">Any</Class>
      <Model xsi:type="xsd:unsignedInt">255</Model>
      <Status xsi:type="xsd:string">Registered</Status>
      <NodeName xsi:type="xsd:string" xsi:nil="true"/>
      <SelectBy xsi:type="xsd:string">Name</SelectBy>
      <SelectItems soapenc:arrayType="ns2:SelectItem[1]" xsi:type="soapenc:Array">
        <item href="#id1"/>
      </SelectItems>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:SelectItem" xmlns:ns3="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Item xsi:type="xsd:string">*</Item>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:SelectCmDeviceResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SelectCmDeviceResult xsi:type="ns1:SelectCmDeviceResult">
        <TotalDevicesFound xsi:type="xsd:unsignedInt">4</TotalDevicesFound>
        <CmNodes soapenc:arrayType="ns1:CmNode[1]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns1:CmNode">
            <ReturnCode xsi:type="ns1:RisReturnCode">Ok</ReturnCode>
            <Name xsi:type="xsd:string">CISCART15</Name>
            <NoChange xsi:type="xsd:boolean">false</NoChange>
            <CmDevices soapenc:arrayType="ns1:CmDevice[4]" xsi:type="soapenc:Array">
              <item xsi:type="ns1:CmDevice">
                <Name xsi:type="xsd:string">ANN_2</Name>
                <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
                <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
```

```
    <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
    <Model xsi:type="xsd:unsignedInt">126</Model>
    <Product xsi:type="xsd:unsignedInt">89</Product>
    <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
    <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
    <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
    <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
    <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
    <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
    <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
    <PerfMonObject xsi:type="xsd:unsignedInt">608</PerfMonObject>
    <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
    <Description xsi:type="xsd:string">ANN_CISCART15</Description>
    <H323Trunk xsi:type="ns1:H323Trunk">
      <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
      <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
      <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
      <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
    </H323Trunk>
   <SIPStatus xsi:type='tns:SIPStatus'>InService</SIPStatus>
   <SIPRemoteIpAddress xsi:type='xsd:string'> 0.77.31.18=Available</SIPRemoteIpAddress>
    <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
  </item>
  <item xsi:type="ns1:CmDevice">
    <Name xsi:type="xsd:string">CFB_2</Name>
    <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
    <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
    <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
    <Model xsi:type="xsd:unsignedInt">50</Model>
    <Product xsi:type="xsd:unsignedInt">28</Product>
    <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
    <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
    <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
    <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
    <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
    <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
    <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
    <PerfMonObject xsi:type="xsd:unsignedInt">15</PerfMonObject>
    <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
    <Description xsi:type="xsd:string">CFB_CISCART15</Description>
    <H323Trunk xsi:type="ns1:H323Trunk">
      <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
      <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
      <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
      <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
    </H323Trunk>
    <TimeStamp xsi:type="xsd:unsignedInt">1204679736</TimeStamp>
  </item>
  <item xsi:type="ns1:CmDevice">
    <Name xsi:type="xsd:string">MOH_2</Name>
    <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
    <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
```

```
            <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
            <Model xsi:type="xsd:unsignedInt">70</Model>
            <Product xsi:type="xsd:unsignedInt">51</Product>
            <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
            <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
            <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
            <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
            <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
            <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
            <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
            <PerfMonObject xsi:type="xsd:unsignedInt">6</PerfMonObject>
            <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
            <Description xsi:type="xsd:string">MOH_CISCART15</Description>
            <H323Trunk xsi:type="ns1:H323Trunk">
              <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
              <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
              <Zone xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
              <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
              <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
              <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
              <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
            </H323Trunk>
            <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
          </item>
          <item xsi:type="ns1:CmDevice">
            <Name xsi:type="xsd:string">MTP_2</Name>
            <IpAddress xsi:type="xsd:string">10.77.31.15</IpAddress>
            <DirNumber xsi:type="xsd:string" xsi:nil="true"/>
            <Class xsi:type="ns1:DeviceClass">MediaResources</Class>
            <Model xsi:type="xsd:unsignedInt">110</Model>
            <Product xsi:type="xsd:unsignedInt">30</Product>
            <BoxProduct xsi:type="xsd:unsignedInt">0</BoxProduct>
            <Httpd xsi:type="ns1:CmDevHttpd">No</Httpd>
            <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
            <IsCtiControllable xsi:type="xsd:boolean">false</IsCtiControllable>
            <LoginUserId xsi:type="xsd:string" xsi:nil="true"/>
            <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
            <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
            <PerfMonObject xsi:type="xsd:unsignedInt">13</PerfMonObject>
            <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
            <Description xsi:type="xsd:string">MTP_CISCART15</Description>
            <H323Trunk xsi:type="ns1:H323Trunk">
              <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
              <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
              <Zone xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
              <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
              <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
              <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
              <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
              <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
            </H323Trunk>
            <TimeStamp xsi:type="xsd:unsignedInt">1204679735</TimeStamp>
          </item>
        </CmDevices>
      </item>
    </CmNodes>
  </SelectCmDeviceResult>
```

```
     <StateInfo xsi:type="xsd:string">&lt;StateInfo ClusterWide=&quot;1&quot;&gt;&lt;Node
Name=&quot;CISCART15&quot; SubsystemStartTime=&quot;1204679712&quot; StateId=&quot;4&quot;
TotalItemsFound=&quot;4&quot;
TotalItemsReturned=&quot;4&quot;/&gt;&lt;/StateInfo&gt;</StateInfo>
    </ns1:SelectCmDeviceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Faults

For details about all the possible faults for SelectCmDevice operation, see SOAP Fault Error Codes, page 4-144.

RisPort Service: selectCtiItem Operation

The selectCtiItem operation allows clients to perform a CTI manager-related query.

Request Format

SOAP Action

The HTTP header should have following SOAP action:

```
SOAPAction: http://schemas.cisco.com/ast/soap/action/#RisPort#SelectCtiItems
```

Envelope Information

The query information should have an Envelope as follows:

```
?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
           xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://schemas.cisco.com/ast/soap/"
xmlns:types="http://schemas.cisco.com/ast/soap/encodedTypes"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

Session ID

The following example shows a SOAP header that contains a session ID. The client sets a unique session ID.

```
<soap:Header>
<tns:AstHeader id="id1">
<SessionId xsi:type="xsd:string">jSessionId</SessionId>
</tns:AstHeader>
</soap:Header>

<soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<tns:SelectCtiItem><StateInfo xsi:type="xsd:string" /><CtiSelectionCriteria href="#id1"
/></tns:SelectCtiItem>
<tns:CtiSelectionCriteria id="id1" xsi:type="tns:CtiSelectionCriteria">
```

Maximum Device Information

The following example specifies the maximum number of devices that this search needs to return:

```
<MaxReturnedItems xsi:type="xsd:unsignedInt">10</MaxReturnedItems>
```

CTI Application/Device/Line Specification

The following example specifies on which CTI manager class Line/Device/Provider a search is provided:

```
<CtiMgrClass xsi:type="tns:CtiMgrClass">Line</CtiMgrClass>
```

Status of CTI Item Search

The following example specifies the Status of class on which to search:

```
<Status xsi:type="tns:CtiStatus">Any</Status>
```

Server Name for Search

The following example specifies the server name on which the search is performed:

```
<NodeName xsi:type="xsd:string" />
```

Type of Search

The following example specifies the type of selection:

```
<SelectAppBy xsi:type="tns:CtiSelectAppBy">AppIpAddress</SelectAppBy>
```

List of Items That Needs to be Searched

The following example specifies an array for items for which the real-time status is required:

```
<AppItems href="#id2" />Name /IP</tns:CtiSelectionCriteria>
<soapenc:Array id="id2" soapenc:arrayType="tns:SelectAppItem[2]">
<Item href="#id3" /><Item xsi:null="1" /></soapenc:Array>
<tns:SelectAppItem id="id3" xsi:type="tns:SelectAppItem">
<AppItem xsi:type="xsd:string"/>
</tns:SelectAppItem>
</soap:Body>
</soap:Envelope>
```

Response Format

The Response includes a sequence of Unified CM Nodes with sequences of CTI devices and lines real-time information:

```
<complexType name='CtiItem'>
<sequence>
<element name='AppId' type='xsd:string'/>
<element name='ProviderName' type='xsd:string'/>
<element name='UserId' type='xsd:string'/>
<element name='AppIpAddr' type='xsd:string'/>
<element name='AppStatus' type='tns:CtiStatus'/>
<element name='AppStatusReason' type='xsd:unsignedInt'/>
<element name='AppTimeStamp' type='xsd:unsignedInt'/>
<element name='CtiDevice' type='tns:CtiDevice'/>
<element name='CtiLine' type='tns:CtiLine'/>
</sequence>
</complexType>
```

CTI Device real-time information contains the following sequence of information:

```
<complexType name='CtiDevice'>
<sequence>
<element name='AppControlsMedia'        type='xsd:boolean'/>
<element name='DeviceName'              type='xsd:string'/>
<element name='DeviceStatus'            type='tns:CtiStatus'/>
<element name='DeviceStatusReason'      type='xsd:unsignedInt'/>
<element name='DeviceTimeStamp'         type='xsd:unsignedInt'/>
</sequence>
</complexType>
```

CTI Line contains the following sequence of information:

```
<complexType name='CtiLine'>
    <sequence>
         <element name='DirNumber'        type='xsd:string'/>
         <element name='LineStatus'       type='tns:CtiStatus'/>
         <element name='LineStatusReason' type='xsd:unsignedInt'/>
         <element name='LineTimeStamp'    type='xsd:unsignedInt'/>
   </sequence>
</complexType>
```

RisPort Service: SelectCtiDevice Operation (Includes IPv6 Devices)

SelectCtiDevice API is used for querying CTI information on CTI application or device or line published from Cisco CTI Manager. The SOAP API supports Call Manager device search for both CTI device with IPv4 and IPv6 addresses. The device search criteria can be either CTI device IPv4 or IPv6 address but not both.

The operation name for invoking the API is **SelectCtiDevice** and the service URL is *https://<server>:8443/realtimeservice/services/RisPort70*

Note The service URL for release 7.1(2) is different from the earlier releases. The service URL of this release includes the version information.

The **SelectCtiDevice** operation comprises the **SelectCtiItemInput** and **SelectCtiItemOutput** messages:

```
<message name="SelectCtiItemInput">
    <part name="StateInfo" type="xsd:string"/>
    <part name="CtiSelectionCriteria" type="tns:CtiSelectionCriteria"/>
</message>
<message name="SelectCtiItemOutput">
    <part name="StateInfo" type="xsd:string"/>
    <part name="SelectCtiItemResult" type="tns:SelectCtiItemResult"/>
</message>
    <portType name="RisPortType">
           <operation name="SelectCtiItem">
           <input message="tns:SelectCtiItemInput"/>
           <output message="tns:SelectCtiItemOutput"/>
       </operation>
    </portType>
```

Request Format

SOAP Action

HTTP header should have following SOAP action for these queries.

```
SOAPAction: http://schemas.cisco.com/ast/soap/action/#RisPort70#SelectCtiDevice
```

Envelope and Session ID

Query information includes an envelope and session ID information as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
<soapenv:Header>
<AstHeader xsi:type="soap:AstHeader">
<SessionId xsi:type="xsd:string"/>
</AstHeader>
</soapenv:Header>
```

The SOAP header has the session ID that is a unique ID from the client.

SelectCtiItem Operation

The SelectCtiItem operation is first defined in the SOAP body element.

```
<ns1:SelectCtiItem soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
```

Stateinfo

If same Information is queried repetitively, then **Stateinfo** is sent from the previous request. **Stateinfo** is the string that is returned by the server and it represents the state of the real-time database.

```
<StateInfo xsi:type="xsd:string" />
```

Selection Criteria

CtiSelectionCriteria is the selection criteria element. You can specify the following:

- MaxReturnedItems
- CtiMgrClass
- Status
- NodeName
- SelectAppBy
- AppItems
- DevNames
- DirNumbers

The format is as follows:

```
<complexType name="CtiSelectionCriteria">
<sequence>
<element name="MaxReturnedItems" nillable="true" type="xsd:unsignedInt"/>
<element name="CtiMgrClass" nillable="true" type="tns:CtiMgrClass"/>
<element name="Status" nillable="true" type="tns:CtiStatus"/>
<element name="NodeName" nillable="true" type="xsd:string"/>
<element name="SelectAppBy" nillable="true" type="tns:CtiSelectAppBy"/>
<element name="AppItems" nillable="true" type="tns:SelectAppItems"/>
<element name="DevNames" nillable="true" type="tns:SelectDevNames"/>
<element name="DirNumbers" nillable="true" type="tns:SelectDirNumbers"/>
</sequence>
</complexType>
```

MaxReturnedItems

Specifies the maximum numnber of items to be returned.

Format:

```
<element name="MaxReturnedItems" nillable="true" type="xsd:unsignedInt"/>
```

CtiMgrClass

Specifies on which CTI manager class line or device, or provider the search is provided. The following values are available:

- Provider
- Device
- Line

Format:

```
<simpleType name="CtiMgrClass">
<restriction base="string">
<enumeration value="Provider"/>
<enumeration value="Device"/>
```

```
<enumeration value="Line"/>
</restriction>
</simpleType>
```

Status

Specifies the status of class on which to search. The following values are available:

- Any
- Open
- Closed
- OpenFailed
- Unknown

Format:

```
<simpleType name="CtiStatus">
<restriction base="string">
<enumeration value="Any"/>
<enumeration value="Open"/>
<enumeration value="Closed"/>
<enumeration value="OpenFailed"/>
<enumeration value="Unknown"/>
</restriction>
</simpleType>
```

NodeName

Specifies the server name on which the search is performed.

Format:

```
<element name="NodeName" nillable="true" type="xsd:string"/>
```

SelectAppBy

Specifies the type of selection. The following options are available:

- AppId
- AppIPV4Address
- AppIPV6Address
- UserId

Format:

```
<simpleType name="CtiSelectAppBy">
<restriction base="string">
<enumeration value="AppId"/>
<enumeration value="AppIPV4Address"/>
<enumeration value="AppIPV6Address"/>
<enumeration value="UserId"/>
</restriction>
</simpleType>
```

Example:

```
<SelectAppBy xsi:type="ns2:CtiSelectAppBy" href="#id2"/>
<multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns6:CtiSelectAppBy"
xmlns:ns5="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">AppId</multiRef>
```

AppItems

Specifies an array for items for which the real-time status is required.

Format:

```
<complexType name="SelectAppItems">
<complexContent>
<restriction base="SOAP-ENC:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:SelectAppItem[]"/>
</restriction>
</complexContent>
</complexType>

<complexType name="SelectAppItem">
<sequence>
<element name="AppItem" type="xsd:string"/>
</sequence>
</complexType>
```

Example:

```
<AppItems soapenc:arrayType="ns2:SelectAppItem[1]" xsi:type="soapenc:Array">
<item href="#id1"/>
</AppItems>
<multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns4:SelectAppItem"
xmlns:ns4="http://ccm.cisco.com/serviceability/soap/risport70/">
<AppItem xsi:type="xsd:string">Quality Report Tool</AppItem>
</multiRef>
```

DevNames

Format:

```
<complexType name="SelectDevNames">
<complexContent>
<restriction base="SOAP-ENC:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:SelectDevName[]"/>
</restriction>
</complexContent>
</complexType>

<complexType name="SelectDevName">
<sequence>
<element name="DevName" type="xsd:string"/>
</sequence>
</complexType>
```

Example:

```
<DevNames xsi:type="ris:SelectDevNames" soapenc:arrayType="ris:SelectDevName[]">
<Item xsi:type="ris:DevName"> dev1 </Item>
```

DirNumbers

Specifies the directory number.

Format:

```
<complexType name="SelectDirNumbers">
<complexContent>
<restriction base="SOAP-ENC:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:SelectDirNumber[]"/>
</restriction>
</complexContent>
</complexType>
```

```
<complexType name="SelectDirNumber">
<sequence>
<element name="DirNumber" type="xsd:string"/>
</sequence>
</complexType>
```

Example:

```
<DirNumbers xsi:type="ns2:SelectDirNumber" xsi:nil="true"/>
```

As part of CtiSelectionCriteria object, the element **CtiSelectAppBy** can have IPv4 or IPv6 addresses as search criteria. In the request, we can specify multiple AppItems, DevNames and DirNumbers. The format of the array elements with such a request is as follows:

```
<AppItems xsi:type= "ris:SelectAppItem" soapenc:arrayType="ris:SelectAppItems[]">
<item xsi:type="ris:AppItem">A Cisco DB</item>
<item xsi:type="ris:AppItem">Cisco</item>
</AppItems>
<DevNames xsi:type="ris:SelectDevNames" soapenc:arrayType="ris:SelectDevName[]">
<Item xsi:type="ris:DevName"> dev1 </Item>
<Item xsi:type="ris:DevName"> dev2 </Item>
</DevNames>

<DirNumbers xsi:type="ris:SelectDirNumbers" soapenc:arrayType="ris:SelectDirNumber[]">
<Item xsi:type="ris:DirNumber"> dir1 </Item>
<Item xsi:type="ris:DirNumber"> dir2 </Item>
</DirNumbers>
```

In the sample request, the ReturnCode is a string:

```
<simpleType name="ReturnCode">
    <restriction base="xsd:string"/>
</simpleType>
```

SelectCtiItem API can be used to search the CtiLine items. In order to do that, the query needs to be modified by having *<CtiMgrClass xsi:type="ris:CtiMgrClass">Line </CtiMgrClass>* in the request.

Response Format

The response follows the following schema:

```
<complexType name="SelectCtiItemResult">
<sequence>
<element name="TotalItemsFound" type="xsd:unsignedInt"/>
<element name="CtiNodes" nillable="true" type="tns:CtiNodes"/>
</sequence>
</complexType>
```

TotalItemsFound

Displays the total items found.

Format:

```
<element name="TotalItemsFound" type="xsd:unsignedInt"/>
```

Example:

```
<TotalItemsFound xsi:type="xsd:unsignedInt">1</TotalItemsFound>
```

Node Information

The following node information are displayed:

- ReturnCode
- Name
- NoChange

- CtiItems

Format:

```
<complexType name="CtiNodes">
<complexContent>
<restriction base="SOAP-ENC:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns:CtiNode[]"/>
</restriction>
</complexContent>
</complexType>

<complexType name="CtiNode">
<sequence>
<element name="ReturnCode" type="tns:RisReturnCode"/>
<element name="Name" type="xsd:string"/>
<element name="NoChange" type="xsd:boolean"/>
<element name="CtiItems" nillable="true" type="tns:CtiItems"/>
</sequence>
</complexType>
```

ReturnCode

Displays the the RIS return codes. The following options are available:

- Ok
- NotFound
- InvalidRequest
- InternalRequest
- InternalError
- NodeNotResponding
- InvalidNodeName

Format:

```
<element name='ReturnCode' type='tns:RisReturnCode'/>
<simpleType name="RisReturnCode">
<restriction base="string">
<enumeration value="Ok"/>
<enumeration value="NotFound"/>
<enumeration value="InvalidRequest"/>
<enumeration value="InternalError"/>
<enumeration value="NodeNotResponding"/>
<enumeration value="InvalidNodeName"/>
</restriction>
```

Name

Displays the name of the node.

Example:

```
Name xsi:type="xsd:string">172.27.203.17</Name>
```

NoChange

Example:

```
<NoChange xsi:type="xsd:boolean">false</NoChange>
```

CtiItems

Displays the following informations:

- AppId
- UserId
- AppIpAddr
- AppIpv6Addr
- AppStatus
- AppStatusReason
- AppTimeStamp
- CtiDevice
- CtiLine

Format:

```
<complexType name="CtiItem">
<sequence>
<element name="AppId" nillable="true" type="xsd:string"/>
<element name="UserId" nillable="true" type="xsd:string"/>
<element name="AppIpAddr" nillable="true" type="xsd:string"/>
<element name="AppIpv6Addr" nillable="true" type="xsd:string"/>
<element name="AppStatus" nillable="true" type="tns:CtiStatus"/>
<element name="AppStatusReason" nillable="true" type="xsd:unsignedInt"/>
<element name="AppTimeStamp" nillable="true" type="xsd:unsignedInt"/>
<element name="CtiDevice" nillable="true" type="tns:CtiDevice"/>
<element name="CtiLine" nillable="true" type="tns:CtiLine"/>
</sequence>
</complexType>
```

AppId

Displays the application ID.

Example:

```
<AppId xsi:type="xsd:string">Quality Report Tool</AppId>
```

UserId

Displays the user ID.

Example:

```
<UserId xsi:type="xsd:string">XXXSysUser</UserId>
```

AppIpAddr

Displays the IPv4 address.

Example:

```
<AppIpAddr xsi:type="xsd:string">xxx.x.x.x</AppIpAddr>
```

AppIpv6Addr

Displays the IPv6 address.

AppStatus

Example:

```
<AppStatus xsi:type="ns2:CtiStatus">Open</AppStatus>
```

AppStatusReason

Example:

```
<AppStatusReason xsi:type="xsd:unsignedInt">0</AppStatusReason>
```

AppTimeStamp

Example:

```
<AppTimeStamp xsi:type="xsd:unsignedInt">1221380548</AppTimeStamp>
```

CtiDevice

Displays the following information realted to CTI device:

- AppControlsMedia
- DeviceName
- DeviceStatus
- DeviceStatusReason
- DeviceTimeStamp

Format:

```
<complexType name="CtiDevice">
<sequence>
<element name="AppControlsMedia" nillable="true" type="xsd:boolean"/>
<element name="DeviceName" nillable="true" type="xsd:string"/>
<element name="DeviceStatus" nillable="true" type="tns:CtiStatus"/>
<element name="DeviceStatusReason" nillable="true" type="xsd:unsignedInt"/>
<element name="DeviceTimeStamp" nillable="true" type="xsd:unsignedInt"/>
</sequence>
</complexType>
```

CtiLine

Displays the following information related to CTI line:

- DirNumber
- LineStatus
- LineStatusReason
- LineTimeStamp

Format:

```
<complexType name="CtiLine">
<sequence>
<element name="DirNumber" type="xsd:string"/>
<element name="LineStatus" type="tns:CtiStatus"/>
<element name="LineStatusReason" type="xsd:unsignedInt"/>
<element name="LineTimeStamp" type="xsd:unsignedInt"/>
</sequence>
</complexType>
```

Request Example

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ris="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
  <soapenv:Body>
    <ns1:SelectCtiItem soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <StateInfo xsi:type="xsd:string" xsi:nil="true"/>
      <CtiSelectionCriteria href="#id0"/>
    </ns1:SelectCtiItem>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:CtiSelectionCriteria" xmlns:ns2="http://schemas.cisco.com/ast/soap/">
      <MaxReturnedItems xsi:type="xsd:unsignedInt">20</MaxReturnedItems>
      <CtiMgrClass xsi:type="ris:CtiMgrClass">Provider</CtiMgrClass>
      <Status xsi:type="ris:CtiStatus">Any</Status>
      <NodeName xsi:type="xsd:string">172.27.203.17</NodeName>
      <SelectAppBy xsi:type="ns2:CtiSelectAppBy" href="#id2"/>
      <AppItems soapenc:arrayType="ns2:SelectAppItem[1]" xsi:type="soapenc:Array">
        <item href="#id1"/>
      </AppItems>
      <DevNames xsi:type="ns2:SelectDevName" xsi:nil="true"/>
      <DirNumbers xsi:type="ns2:SelectDirNumber" xsi:nil="true"/>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns4:SelectAppItem"
xmlns:ns4="http://ccm.cisco.com/serviceability/soap/risport70/">
      <AppItem xsi:type="xsd:string">Quality Report Tool</AppItem>
    </multiRef>
    <multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns6:CtiSelectAppBy"
xmlns:ns5="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">AppId</multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <ns1:SelectCtiItemResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
         <SelectCtiItemResult xsi:type="ns2:SelectCtiItemResult"
xmlns:ns2="http://schemas.cisco.com/ast/soap/risport70/">
            <TotalItemsFound xsi:type="xsd:unsignedInt">1</TotalItemsFound>
            <CtiNodes soapenc:arrayType="ns3:CtiNode[1]" xsi:type="soapenc:Array"
xmlns:ns3="http://ccm.cisco.com/serviceability/soap/risport70/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
               <item xsi:type="ns2:CtiNode">
                  <ReturnCode xsi:type="ns3:RisReturnCode">Ok</ReturnCode>
                  <Name xsi:type="xsd:string">172.27.203.17</Name>
                  <NoChange xsi:type="xsd:boolean">false</NoChange>
                  <CtiItems soapenc:arrayType="ns3:CtiItem[1]" xsi:type="soapenc:Array">
```

```
                        <item xsi:type="ns2:CtiItem">
                           <AppId xsi:type="xsd:string">Quality Report Tool</AppId>
                           <UserId xsi:type="xsd:string">CCMQRTSysUser</UserId>
                           <AppIpAddr xsi:type="xsd:string">127.0.0.1</AppIpAddr>
                           <AppIpv6Addr xsi:type="xsd:string" xsi:nil="true"/>
                           <AppStatus xsi:type="ns2:CtiStatus">Open</AppStatus>
                           <AppStatusReason xsi:type="xsd:unsignedInt">0</AppStatusReason>
                           <AppTimeStamp xsi:type="xsd:unsignedInt">1221380548</AppTimeStamp>
                           <CtiDevice xsi:type="ns3:CtiDevice" xsi:nil="true"/>
                           <CtiLine xsi:type="ns3:CtiLine" xsi:nil="true"/>
                        </item>
                     </CtiItems>
                  </item>
               </CtiNodes>
            </SelectCtiItemResult>
            <StateInfo xsi:type="xsd:string">&lt;StateInfo>&lt;Node Name="172.27.203.17"
SubsystemStartTime="1221380428" StateId="134" TotalItemsFound="1"
TotalItemsReturned="1"/>&lt;/StateInfo></StateInfo>
         </ns1:SelectCtiItemResponse>
      </soapenv:Body>
</soapenv:Envelope>
```

Fault

For details about all the possible faults for SelectCmDevice operation, see SOAP Fault Error Codes, page 4-144.

RisPort Service: getServerInfo Operation

The getServerInfo operation exports the following information from the Server Information SOAP interface:

- Host Name =MCS-SD4
- OS Name =Linux
- OS Arch =i386
- OS Version =2.4.21-15.ELsmp
- Java Runtime Version =1.4.2_05-b04
- Java Virtual Machine vendor =Sun Microsystems Inc.
- CallManager Version =7.0.1

The getServerInfo operation comprises the getServerInfoRequest and getServerInfoResponse:

```
<wsdl:operation name="getServerInfo">
  <wsdlsoap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#PerfmonPort#GetServerInfo" />
<wsdl:input name="getServerInfoRequest">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://schemas.cisco.com/ast/soap/" use="encoded" />
  </wsdl:input>
<wsdl:output name="getServerInfoResponse">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://schemas.cisco.com/ast/soap/" use="encoded" />
  </wsdl:output>
  </wsdl:operation>
```

Request Format

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://schemas.cisco.com/ast/soap/">
   <soapenv:Header>
      <AstHeader xsi:type="soap:AstHeader">
         <SessionId xsi:type="xsd:string">999</SessionId>
      </AstHeader>
   </soapenv:Header>
   <soapenv:Body>
      <soap:GetServerInfo
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <Hosts xsi:type="soap:ArrayOfHosts">
            <!--Zero or more repetitions:-->
            <item xsi:type="xsd:string">UCM-HostNameorIP</item>
         </Hosts>
      </soap:GetServerInfo>
   </soapenv:Body>
</soapenv:Envelope>
```

In the request, an ArrayOfHosts definition gets specified, and the response provides the server information for the list of hostnames that are specified in the array.

```
<wsdl:message name="getServerInfoRequest">
  <wsdl:part name="Hosts" type="impl:ArrayOfHosts" />
</wsdl:message>
```

Response Format

The response comprises an ArrayOfServerInfo:

```
<wsdl:message name="getServerInfoResponse">
  <wsdl:part name="ServerInfo" type="impl:ArrayOfServerInfo" />
  </wsdl:message>
<complexType name="ArrayOfServerInfo">
<complexContent>
<restriction base="soapenc:Array">
  <attribute ref="soapenc:arrayType" wsdl:arrayType="impl:ServerInformation[]" />
  </restriction>
  </complexContent>
  </complexType>
```

ServerInformation consists of the following sequence of elements:

```
<complexType name="ServerInformation">
<sequence>
  <element name="HostName" nillable="true" type="xsd:string" />
  <element name="os-name" nillable="true" type="xsd:string" />
  <element name="os-version" nillable="true" type="xsd:string" />
  <element name="os-arch" nillable="true" type="xsd:string" />
  <element name="java-runtime-version" nillable="true" type="xsd:string" />
  <element name="java-vm-vendor" nillable="true" type="xsd:string" />
  <element name="call-manager-version" nillable="true" type="xsd:string" />
  <element name="Active-versions" nillable="true" type="xsd:string" />
   <element name="InActive-versions" nillable="true" type="xsd:string" />
  </sequence>
  </complexType>
```

Faults

The Server sends a fault for “Error message context is NULL.” This error does not appear in normal operation.

The following fault is sent if an HTTPS connection fails to a remote server — “Error initiating https connection to <URL>.”

Example

Request example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetServerInfo soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <Hosts soapenc:arrayType="soapenc:string[1]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="soapenc:string">10.77.31.15</item>
      </Hosts>
    </ns1:GetServerInfo>
  </soapenv:Body>
</soapenv:Envelope>
```

Response example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetServerInfoResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServerInfo soapenc:arrayType="ns1:ServerInformation[1]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:ServerInformation">
          <HostName xsi:type="xsd:string">CISCART15</HostName>
          <os-name xsi:type="xsd:string">VOS</os-name>
          <os-version xsi:type="xsd:string">2.6.9-42.ELsmp</os-version>
          <os-arch xsi:type="xsd:string">i386</os-arch>
          <java-runtime-version xsi:type="xsd:string">1.5.0_14-b03</java-runtime-version>
          <java-vm-vendor xsi:type="xsd:string">Sun Microsystems Inc.</java-vm-vendor>
          <call-manager-version xsi:type="xsd:string">7.0.0.39700-8</call-manager-version>
         <Active_Versions xsi:type="xsd:string">hwdata-0.146.23.EL-1 : os-ver-4.0.0.0-7 :
redhat-logos-1.1.26-1 : setup-2.5.37-1.3 : basesystem-8.0-4 : tzdata-2007k-1.el4 :
glibc-2.3.4-2.25 : beecrypt-3.1.0-6 : chkconfig-1.3.13.4-1 :
compat-libstdc++-296-2.96-132.7.2 : dos2unix-3.1-21.2 : e2fsprogs-1.35-12.4.EL4 :
elfutils-libelf-0.97.1-3 : ethtool-1.8-4 : gdbm-1.8.0-24 : glib2-2.4.7-1 :
iputils-20020927-18.EL4.3 : attr-2.4.16-3 : acl-2.2.23-5 : libgpg-error-1.0-1 :
libselinux-1.19.1-7.2 : device-mapper-1.02.07-4.0.RHEL4 : db4-4.2.52-7.1 :
libtermcap-2.0.8-39 : mingetty-1.07-3 : bash-3.0-19.3 : bzip2-1.0.2-13.EL4.3 :
fior-0.99.1-2.el4 : iproute-2.6.9-3 : mt-st-0.8-1 : ncurses-5.4-13 :
net-tools-1.60-37.EL4.8 : openssl096b-0.9.6b-22.46 : pcre-4.5-3.2.RHEL4 :
perl-Filter-1.30-6 : logrotate-3.7.1-5.RHEL4 : redhat-release-4AS-5.5 : schedutils-1.4.0-2
: setserial-2.17-17 : slang-1.4.9-8 : snmp-mon-0.0.0.19-9 : strace-4.5.14-0.EL4.1 :
expect-5.42.1-1 : tmpwatch-2.9.1-1 : unzip-5.51-7 : vim-minimal-6.3.046-0.40E.7 :
zip-2.3-27 : file-4.10-2.EL4.4 : binutils-2.15.92.0.2-21 : diffutils-2.8.1-12 :
gawk-3.1.3-10.1 : grep-2.5.1-32.2 : ash-0.3.8-20 : gzip-1.3.3-16.rhel4 :
krb5-libs-1.3.4-27 : libidn-0.5.6-1 : libxslt-1.1.11-1 : mgetty-1.1.31-2 :
openssl-0.9.7a-43.14 : bind-utils-9.2.4-24.EL4 : net-snmp-libs-5.1.2-11.EL4.7 :
pdksh-5.2.14-30.3 : readline-4.3-13 : lvm2-2.02.06-6.0.RHEL4 : libxml2-python-2.6.16-6 :
rpm-libs-4.3.3-18_nonptl : shadow-utils-4.0.3-60.RHEL4 : dbus-glib-0.22-12.EL.8 :
nscd-2.3.4-2.25 : rpm-4.3.3-18_nonptl : sysklogd-1.4.1-26_EL : sysreport-1.3.15-8 :
tftp-0.39-1 : tomcat-5.5.17-0 : cactus-12.1.5-3 : elixir-4.2-3 : log4j-1.2.8-3 :
pgjdbc-7.3.3-3 : saaj-1.3-3 : unixODBC-2.2.11-1.RHEL4.1 : vim-common-6.3.046-0.40E.7 :
xerces-2.6.2-3 : cracklib-2.7-29 : pam-0.77-66.17 : authconfig-4.6.10-rhel4.3 :
policycoreutils-1.18.1-4.9 : sudo-1.6.7p5-30.1.3 : util-linux-2.12a-16.EL4.20 :
udev-039-10.15.EL4 : initscripts-7.93.25.EL-1 : cyrus-sasl-2.1.19-5.EL4 :
dhclient-3.0.1-58.EL4 : kbd-1.12-2 : kernel-utils-2.4-13.1.83 : mkinitrd-4.2.1.8-1 :
kernel-smp-2.6.9-42.EL : iptables-1.2.11-3.1.RHEL4 : libpcap-0.8.3-10.RHEL4 :
net-snmp-utils-5.1.2-11.EL4.7 : gnupg-1.2.6-9 : nss_ldap-226-17 :
openssh-clients-3.9p1-8.RHEL4.17.1 : openssh-server-3.9p1-8.RHEL4.17.1 : passwd-0.68-10.1
: tcpdump-3.8.2-10.RHEL4 : sysstat-5.0.5-11.rhel4 : master-7.0.0.39700-8 :
platform-script-2.0.0.1-1 : platform-common-2.0.0.1-1 : platform-drf-2.0.0.2-2 :
platform-servM-2.0.0.1-1 : platform-ipsec-1.0.0.0-1 : platform-api-3.0.0.0-6 :
platform-util-2.0.0.1-1 : cm-script-5.0.1.0-1 : cm-lib-1.0.0.0-1 : cm-dbms-1.0.0.0-1 :
cm-ccmadmin-1.1.0.0-1 : cm-bps-1.1.0.0-1 : cm-alarm-0.0.0.1-0 : cm-ccmmib-0.0.0.1-0 :
cm-cdp-0.0.0.1-0 : cm-svc-web-0.0.0.1-1 : cm-RIS-0.0.0.1-0 : cm-reporter-0.0.0.1-0 :
cm-rtmt-client-plugin-0.0.0.2-0 : cm-soap-cdrondemandservice-0.0.0.1-0 :
cm-soap-perfmonservice-0.0.0.1-0 : cm-car-1.0.0.0-1 : cm-security-1.0.0.0-1 :
cm-ctlp-1.0.0.0-1 : cm-dna-5.0.0.2-1 : cm-authentication-1.0.0.1-0 : cm-dirsync-1.0.0.1-0
: cm-ac-2.0.0.1-0 : cm-locale-english_united_states-7.1.0.1-1 : cm-axl-1.1.0.0-1 :
cm-ccmuser-1.1.0.0-1 : cm-ipvms-7.0.0.0-1 : cm-tsp-plugin-7.0.0.2-0 :
cm-app-services-0.0.0.1-0 : cm-webdialer-0.0.0.1-0 : cm-jtapi-plugin-7.0.0.9700-2 :
cm-licensing-2.0.0.0-0 : cm-devicepack-1.0.0.0-0 : cm-ccmhelp-1.1.0.0-2 :
cm-srstctlclient-0.0.0.1-0 : cm-cmtomcat-1.1.0.0-1 : cm-grt-1.1.0.0-1 : msg-2007.10-1 :
IIF-10.00.UC7X1-1 : glsclient-4.00.UC7-1 : hpasm-7.8.0-88.rhel4 : cmanic-7.8.0-3.rhel4 :
```

hpadu-7.80-6 : comps-4AS-0.20060803 : libgcc-3.4.6-3.1 : preferences-1.0.1-6 :
rootfiles-8-1 : filesystem-2.3.0-1 : termcap-5.4-3 : glibc-common-2.3.4-2.25 :
audit-libs-1.0.14-1.EL4 : bzip2-libs-1.0.2-13.EL4.3 : compat-db-4.1.25-9 :
compat-libstdc++-33-3.2.3-47.3 : dosfstools-2.8-15 : eject-2.0.13-11 : elfutils-0.97.1-3 :
expat-1.95.7-4 : glib-1.2.10-15 : hdparm-5.7-2 : libattr-2.4.16-3 : libacl-2.2.23-5 :
libcap-1.10-20 : libgcrypt-1.2.0-3 : libsepol-1.1.1-2 : libstdc++-3.4.6-3.1 : gmp-4.1.4-3
: lsof-4.72-1.4 : mktemp-1.5-20 : audit-1.0.14-1.EL4 : crontabs-1.10-7 : fiostats-0.99.1-9
: java-provides-1.0.0.2-0 : nc-1.10-22 : less-382-4 : OpenIPMI-libs-1.4.14-1.4E.13 :
patch-2.5.4-20 : perl-5.8.5-36.RHEL4 : popt-1.9.1-18_nonptl : psmisc-21.4-4.1 :
rsync-2.6.3-1 : setarch-1.6-1 : sg3_utils-1.06-3 : newt-0.51.6-9.rhel4 : star-1.5a25-6 :
tcl-8.4.7-2 : tcp_wrappers-7.6-37.2 : traceroute-1.4a12-24 : usbutils-0.11-6.1 :
words-3.0-3 : zlib-1.2.1.2-1.2 : info-4.7-5.el4.2 : cpio-2.5-9.RHEL4 :
findutils-4.1.20-7.el4.1 : gdb-6.3.0.0-1.132.EL4 : coreutils-5.2.1-31.4 : grub-0.95-3.5 :
jdk-1.5.0_14-fcs : krb5-workstation-1.3.4-27 : libxml2-2.6.16-6 : make-3.80-6.EL4 :
module-init-tools-3.1-0.pre5.3.2 : bind-libs-9.2.4-24.EL4 : curl-7.12.1-8.rhel4 :
OpenIPMI-1.4.14-1.4E.13 : procps-3.2.3-8.4 : bc-1.06-17.1 : python-2.3.4-14.3 :
PyXML-0.8.3-6 : sed-4.1.2-5.EL4 : dbus-0.22-12.EL.8 : MAKEDEV-3.15.2-3 :
ntp-4.2.0.a.20040617-4.EL4.1 : stunnel-4.05-3 : tar-1.14-12.RHEL4 : tcsh-6.13-9 :
time-1.7-25 : activation-1.0.0-3 : ecs-1.4.2-3 : jaxm-1.1.2-3 : mail-1.0.0-3 :
regexp-1.3-3 : struts-1.1-3 : utempter-0.5.5-5 : xalan-2.7.0-3 : xmlstarlet-1.0.1-1 :
cracklib-dicts-2.7-29 : at-3.1.8-80_EL4 : pam_krb5-2.1.8-1 : screen-4.0.2-5 :
SysVinit-2.85-34.3 : hotplug-2004_04_01-7.7 : hal-0.4.2-4.EL4 : acpid-1.0.3-2 :
cyrus-sasl-md5-2.1.19-5.EL4 : ipsec-tools-0.3.3-6.rhel4.1 : kudzu-1.1.95.15-1 :
lm_sensors-2.8.7-2.40.3 : kernel-2.6.9-42.EL : dhcp-3.0.1-58.EL4 :
iptables-ipv6-1.2.11-3.1.RHEL4 : net-snmp-5.1.2-11.EL4.7 : openldap-2.2.13-6.4E :
libuser-0.52.5-1.el4.1 : openssh-3.9p1-8.RHEL4.17.1 : netdump-0.7.16-2 :
netdump-server-0.7.16-2 : pciutils-2.1.99.test8-3.2 : vixie-cron-4.1-44.EL4 : which-2.16-4
: platform-ver-2.0.0.1-1 : platform-ui-2.0.0.1-1 : platform-licensing-2.0.0.0-1 :
platform-remotesupport-2.0.0.1-3 : platform-cm-1.0.0.0-1 : platform-csa-5.2.0-245.1 :
platform-clm-2.0.0.1-1 : os-services-1.1-1 : cm-ver-7.0.0.39700-8 : cm-pi-0.0.0.1-0 :
cm-dbl-1.0.0.0-1 : cm-cmplatform-1.1.0.0-1 : cm-taps-plugin-7.0.2.0-1 :
cm-syslog-0.0.0.1-0 : cm-sysapp-1.0.0.0-1 : cm-lpm-0.0.0.1-0 :
cm-reporter-servlet-0.0.0.1-0 : cm-amc-0.0.0.1-0 : cm-rtmt-servlet-0.0.0.1-0 :
cm-log4jinit-servlet-0.0.0.1-0 : cm-soap-logcollectionservice-0.0.0.1-0 :
cm-soap-realtimeservice-0.0.0.1-0 : cm-cef-0.0.0.1-0 : cm-cdrdlv-1.0.0.0-1 :
cm-capf-1.0.0.0-1 : cm-ccm-5.0.1.0-0 : cm-encryption-1.0.0.1-0 : cm-scheduler-1.0.0.1-0 :
cm-CTIManager-1.0.0.1-0 : cm-tftp-1.0.0.1-0 : cm-axlsqltoolkit-plugin-1.1.0.0-1 :
cm-pd-1.0.0.0-1 : cm-ccmcip-1.0.0.1-0 : cm-ipvmsd-6.0.0.1-1 : cm-ctlc-plugin-6.0.0.1-1 :
cm-em-0.0.0.1-0 : cm-ipma-0.0.0.1-0 : cm-cmi-1.0.0.1-0 : cm-tct-svc-0.0.0.1-1 :
cm-tomcatstats-0.0.0.1-0 : cm-dhcp-1.0.0.1-0 : cm-cfrt-0.0.0.1-0 : cm-ccmivr-6.0.0.1-1 :
cm-cucreports-1.1.0.0-1 : gls-4.00.UC10-1 : msgclient-2004.3-1 : csdk-2.90.UC4XD-1 :
hprsm-7.8.0-custom : hpsmh-2.1.8-177 : hponcfg-1.6.0-1 :</Active_Versions>
          <In_Active_Versions xsi:type="xsd:string">no packages :</In_Active_Versions>
        </item>
      </ServerInfo>
    </ns1:GetServerInfoResponse>
  </soapenv:Body>
</soapenv:Envelope>

RisPort Service: SelectCmDeviceSIP Operation

The SelectCmDeviceSIP operation allows clients to perform Unified CM SIP device related queries.

Request Format

SOAP Action

The HTTP header contains the following SOAP action for these queries:

```
SOAPAction: "http://schemas.cisco.com/ast/soap/action/#RisPort#SelectCmDeviceSIP"
```

Envelope Information

Query information should have an Envelope as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
    <soapenv:Envelope xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <soapenv:Body>
                    <ns1:SelectCmDeviceSIP
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
```

State Info

If the same information is queried over and over again then Stateinfo needs to be sent from the previous request for each repetitive query by a client.

```
<StateInfo xsi:type="xsd:string"/>
```

The selection criteria type CmSelectionCriteriaSIP follows the optional State Info:

```
<CmSelectionCriteriaSIP href="#id0"/>
</ns1:SelectCmDeviceSIP>
```

CmSelectionCriteriaSIP comprises the following items:

- **MaxReturnedDevices**—Specifies how many maximum devices that can be returned for search criteria

  ```
  <MaxReturnedDevices xsi:type="xsd:unsignedInt">200</MaxReturnedDevices>
  ```

- **Class**—Specifies the device class type that needs to be queried for Real-time status. Device classes are Any, Phone, Gateway, H323, Cti, VoiceMail, MediaResources and Unknown.

  ```
  <Class xsi:type="xsd:string">Any</Class>
  ```

- **Model**—Specifies the model of the device. 255 applies for all models.

  ```
  <Model xsi:type="xsd:unsignedInt">255</Model>
  ```

- **Status**—Specifies the device status in search criteria, which is one of Any, Registered, UnRegistered, Rejected, PartiallyRegistered or Unknown.

  ```
  <Status xsi:type="xsd:string">Registered</Status>
  ```

- **NodeName**—Specifies the server name where the search needs to be performed. If you do not specify a name, the system will search in all servers in cluster.

  ```
  <NodeName xsi:type="xsd:string" xsi:nil="true"/>
  ```

- **SelectBy**—Specifies the selection type for whether it is IP Address/Name.

```
<SelectBy xsi:type="xsd:string">Name</SelectBy>
```

- **SelectItems**—Specifies the array of items for which search criteria is specified. The following specifies an array that contains the IP Address or Device Name of the items for which the real-time status is needed.

```
<SelectItems xsi:type="ns2:SelectItem" xsi:nil="true"/>
```

- **Protocol**—Specifies the protocol name in the search criteria, which is one of Any, SCCP, SIP, or Unknown.

```
<Protocol xsi:type="ns3:Protocol" xsi:nil="true"/>

</soapenv:Body>
</soapenv:Envelope>
```

Response Format

Response follows this schema and contains one to many node information, plus the stateInfo that the SOAP server returns. Each node includes a sequence of search information that was found based on the search criteria.

```
<complexType name='SelectCmDeviceResultSIP'>
   <sequence>
      <element name='TotalDevicesFound' type='xsd:unsignedInt'/>
      <element name='CmNodes' type='tns:CmNodesSIP'/>
   </sequence>
</complexType>
```

CmNodesSIP is list of CmNodeSIP that are given in the search criteria.

```
<complexType name="CmNodesSIP">
  <complexContent>
   <restriction base="SOAP-ENC:Array">
     <attribute ref="soapenc:arrayType" wsdl:arrayType="tns:CmNodeSIP[]"/>
   </restriction>
  </complexContent>
</complexType>
```

Each CmNodesSIP has a sequence of devices and their registration status.

```
<complexType name='CmNodeSIP'>
     <sequence>
         <element name='ReturnCode'          type='tns:RisReturnCode'/>
         <element name='Name'                type='xsd:string'/>
         <element name='NoChange'            type='xsd:boolean'/>
         <element name='CmDevices'           type='tns:CmDevicesSIP'/>
    </sequence>
</complexType>
<complexType name="CmDevicesSIP">
  <complexContent>
   <restriction base="SOAP-ENC:Array">
     <attribute ref="soapenc:arrayType" wsdl:arrayType="tns:CmDeviceSIP[]"/>
   </restriction>
  </complexContent>
</complexType>
```

CmDeviceSIP information will contain the following information:

```
<complexType name="CmDeviceSIP">
    <sequence>
        <element name="Name" type="xsd:string"/>
        <element name="IpAddress" type="xsd:string"/>
```

```
        <element name="DirNumber" type="xsd:string"/>
        <element name="Class" type="tns:DeviceClass"/>
        <element name="Model" type="xsd:unsignedInt"/>
        <element name="Product" type="xsd:unsignedInt"/>
        <element name="BoxProduct" type="xsd:unsignedInt"/>
        <element name="Httpd" type="tns:CmDevHttpd"/>
        <element name="RegistrationAttempts" type="xsd:unsignedInt"/>
        <element name="IsCtiControllable" type="xsd:boolean"/>
        <element name="LoginUserId" type="xsd:string"/>
        <element name="Status" type="tns:CmDevRegStat"/>
        <element name="StatusReason" type="xsd:unsignedInt"/>
        <element name="PerfMonObject" type="xsd:unsignedInt"/>
        <element name="DChannel" type="xsd:unsignedInt"/>
        <element name="Description" type="xsd:string"/>
        <element name="H323Trunk" type="tns:H323Trunk"/>
        <element name="TimeStamp" type="xsd:unsignedInt"/>
        <element name="Protocol" type="tns:ProtocolType"/>
        <element name="NumOfLines" type="xsd:unsignedInt"/>
        <element name="LinesStatus" type="tns:CmDevLinesStatus"/>
    </sequence>
</complexType>
```

Protocol defines the following enumerated protocol types:

```
<simpleType name="ProtocolType">
    <restriction base="string">
        <enumeration value="Any"/>
        <enumeration value="SCCP"/>
        <enumeration value="SIP"/>
        <enumeration value="Unknown"/>
    </restriction>
</simpleType>
```

CmDevLinesStatus is a list of CmDevSingleLineStatus:

```
<complexType name="CmDevLinesStatus">
    <complexContent>
        <restriction base="SOAP-ENC:Array">
            <attribute ref="soapenc:arrayType"
            wsdl:arrayType="tns:CmDevSingleLineStatus[]"/>
        </restriction>
    </complexContent>
</complexType>
```

CmSingleLineStatus is a sequence of DN and DN status:

```
<complexType name="CmDevSingleLineStatus">
    <sequence>
        <element name="DirectoryNumber" type="xsd:string"/>
        <element name="Status" type="tns:CmSingleLineStatus"/>
    </sequence>
</complexType>
```

CmSingleLineStatus defines the enumerated DN status as follows:

```
<simpleType name="CmSingleLineStatus">
    <restriction base="string">
        <enumeration value="Any"/>
        <enumeration value="Registered"/>
        <enumeration value="UnRegistered"/>
        <enumeration value="Rejected"/>
        <enumeration value="Unknown"/>
    </restriction>
</simpleType>
```

Example

The following example shows a SelectCmDeviceSIP response:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:SelectCmDeviceSIPResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <SelectCmDeviceResultSIP xsi:type="ns1:SelectCmDeviceResultSIP">
    <TotalDevicesFound xsi:type="xsd:unsignedInt">4</TotalDevicesFound>
    <CmNodes xsi:type="soapenc:Array" soapenc:arrayType="ns1:CmNodeSIP[2]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>
      <ReturnCode xsi:type="ns1:RisReturnCode">Ok</ReturnCode>
      <Name xsi:type="xsd:string">node70</Name>
      <NoChange xsi:type="xsd:boolean">false</NoChange>
      <CmDevices xsi:type="soapenc:Array" soapenc:arrayType="ns1:CmDeviceSIP[4]">
       <item>
        <Name xsi:type="xsd:string">SEP003094C25B01</Name>
        <IpAddress xsi:type="xsd:string">192.20.0.1</IpAddress>
        <DirNumber xsi:type="xsd:string">5001-Registered</DirNumber>
        <Class xsi:type="ns1:DeviceClass">Phone</Class>
        <Model xsi:type="xsd:unsignedInt">7</Model>
        <Product xsi:type="xsd:unsignedInt">35</Product>
        <BoxProduct xsi:type="xsd:unsignedInt" xsi:nil="true"/>
        <Httpd xsi:type="ns1:CmDevHttpd">Yes</Httpd>
        <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
        <IsCtiControllable xsi:type="xsd:boolean">true</IsCtiControllable>
        <LoginUserId xsi:type="xsd:string">jdas0</LoginUserId>
        <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
        <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
        <PerfMonObject xsi:type="xsd:unsignedInt">2</PerfMonObject>
        <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
        <Description xsi:type="xsd:string">Fake data</Description>
        <H323Trunk xsi:type="ns1:H323Trunk">
         <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
         <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
         <Zone xsi:type="xsd:string" xsi:nil="true"/>
         <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
         <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
         <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
         <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
         <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
         <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
         <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
        </H323Trunk>
        <TimeStamp xsi:type="xsd:unsignedInt">1110841855</TimeStamp>
        <Protocol xsi:type="ns1:ProtocolType">SIP</Protocol>
        <NumOfLines xsi:type="xsd:unsignedInt">1</NumOfLines>
        <LinesStatus xsi:type="soapenc:Array"
               soapenc:arrayType="ns1:CmDevSingleLineStatus[1]">
         <item>
          <DirectoryNumber xsi:type="xsd:string">5001</DirectoryNumber>
          <Status xsi:type="ns1:CmSingleLineStatus">Registered</Status>
         </item>
        </LinesStatus>
       </item>
       <item>
        <Name xsi:type="xsd:string">SEP003094C25B02</Name>
        <IpAddress xsi:type="xsd:string">192.20.0.2</IpAddress>
```

```
 <DirNumber xsi:type="xsd:string">5002-Registered</DirNumber>
 <Class xsi:type="ns1:DeviceClass">Phone</Class>
 <Model xsi:type="xsd:unsignedInt">7</Model>
 <Product xsi:type="xsd:unsignedInt">35</Product>
 <BoxProduct xsi:type="xsd:unsignedInt" xsi:nil="true"/>
 <Httpd xsi:type="ns1:CmDevHttpd">Yes</Httpd>
 <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
 <IsCtiControllable xsi:type="xsd:boolean">true</IsCtiControllable>
 <LoginUserId xsi:type="xsd:string">jdas1</LoginUserId>
 <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
 <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
 <PerfMonObject xsi:type="xsd:unsignedInt">2</PerfMonObject>
 <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
 <Description xsi:type="xsd:string">Fake data</Description>
 <H323Trunk xsi:type="ns1:H323Trunk">
  <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
  <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
  <Zone xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
  <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
       <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
  <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
  <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
 </H323Trunk>
 <TimeStamp xsi:type="xsd:unsignedInt">1110841855</TimeStamp>
 <Protocol xsi:type="ns1:ProtocolType">SIP</Protocol>
 <NumOfLines xsi:type="xsd:unsignedInt">1</NumOfLines>
 <LinesStatus xsi:type="soapenc:Array"
        soapenc:arrayType="ns1:CmDevSingleLineStatus[1]">
  <item>
   <DirectoryNumber xsi:type="xsd:string">5002</DirectoryNumber>
   <Status xsi:type="ns1:CmSingleLineStatus">Registered</Status>
  </item>
 </LinesStatus>
</item>
<item>
 <Name xsi:type="xsd:string">SEP003094C25B03</Name>
 <IpAddress xsi:type="xsd:string">192.20.0.3</IpAddress>
 <DirNumber xsi:type="xsd:string">5003-Registered</DirNumber>
 <Class xsi:type="ns1:DeviceClass">Phone</Class>
 <Model xsi:type="xsd:unsignedInt">7</Model>
 <Product xsi:type="xsd:unsignedInt">35</Product>
 <BoxProduct xsi:type="xsd:unsignedInt" xsi:nil="true"/>
 <Httpd xsi:type="ns1:CmDevHttpd">Yes</Httpd>
 <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
 <IsCtiControllable xsi:type="xsd:boolean">true</IsCtiControllable>
 <LoginUserId xsi:type="xsd:string">jdas2</LoginUserId>
 <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
 <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
 <PerfMonObject xsi:type="xsd:unsignedInt">2</PerfMonObject>
 <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
 <Description xsi:type="xsd:string">Fake data</Description>
 <H323Trunk xsi:type="ns1:H323Trunk">
  <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
  <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
  <Zone xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
  <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
  <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
  <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
  <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
```

```
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
     </H323Trunk>
     <TimeStamp xsi:type="xsd:unsignedInt">1110841855</TimeStamp>
     <Protocol xsi:type="ns1:ProtocolType">SIP</Protocol>
     <NumOfLines xsi:type="xsd:unsignedInt">1</NumOfLines>
     <LinesStatus xsi:type="soapenc:Array"
            soapenc:arrayType="ns1:CmDevSingleLineStatus[1]">
      <item>
       <DirectoryNumber xsi:type="xsd:string">5003</DirectoryNumber>
       <Status xsi:type="ns1:CmSingleLineStatus">Registered</Status>
      </item>
     </LinesStatus>
    </item>
    <item>
     <Name xsi:type="xsd:string">SEP003094C25B04</Name>
     <IpAddress xsi:type="xsd:string">192.20.0.4</IpAddress>
     <DirNumber xsi:type="xsd:string">5004-Registered</DirNumber>
     <Class xsi:type="ns1:DeviceClass">Phone</Class>
     <Model xsi:type="xsd:unsignedInt">7</Model>
     <Product xsi:type="xsd:unsignedInt">35</Product>
     <BoxProduct xsi:type="xsd:unsignedInt" xsi:nil="true"/>
     <Httpd xsi:type="ns1:CmDevHttpd">Yes</Httpd>
     <RegistrationAttempts xsi:type="xsd:unsignedInt">0</RegistrationAttempts>
     <IsCtiControllable xsi:type="xsd:boolean">true</IsCtiControllable>
     <LoginUserId xsi:type="xsd:string">jdas3</LoginUserId>
     <Status xsi:type="ns1:CmDevRegStat">Registered</Status>
     <StatusReason xsi:type="xsd:unsignedInt">0</StatusReason>
     <PerfMonObject xsi:type="xsd:unsignedInt">2</PerfMonObject>
     <DChannel xsi:type="xsd:unsignedInt">0</DChannel>
     <Description xsi:type="xsd:string">Fake data</Description>
     <H323Trunk xsi:type="ns1:H323Trunk">
      <ConfigName xsi:type="xsd:string" xsi:nil="true"/>
      <TechPrefix xsi:type="xsd:string" xsi:nil="true"/>
      <Zone xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer1 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer2 xsi:type="xsd:string" xsi:nil="true"/>
      <RemoteCmServer3 xsi:type="xsd:string" xsi:nil="true"/>
      <AltGkList xsi:type="xsd:string" xsi:nil="true"/>
      <ActiveGk xsi:type="xsd:string" xsi:nil="true"/>
      <CallSignalAddr xsi:type="xsd:string" xsi:nil="true"/>
      <RasAddr xsi:type="xsd:string" xsi:nil="true"/>
     </H323Trunk>
     <TimeStamp xsi:type="xsd:unsignedInt">1110841855</TimeStamp>
     <Protocol xsi:type="ns1:ProtocolType">SIP</Protocol>
     <NumOfLines xsi:type="xsd:unsignedInt">1</NumOfLines>
     <LinesStatus xsi:type="soapenc:Array"
                soapenc:arrayType="ns1:CmDevSingleLineStatus[1]">
      <item>
       <DirectoryNumber xsi:type="xsd:string">5004</DirectoryNumber>
       <Status xsi:type="ns1:CmSingleLineStatus">Registered</Status>
      </item>
     </LinesStatus>
    </item>
   </CmDevices>
  </item>
  <item>
   <ReturnCode xsi:type="ns1:RisReturnCode">NotFound</ReturnCode>
   <Name xsi:type="xsd:string">node71</Name>
   <NoChange xsi:type="xsd:boolean">false</NoChange>
   <CmDevices xsi:type="soapenc:Array" soapenc:arrayType="ns1:CmDeviceSIP[0]"/>
  </item>
 </CmNodes>
</SelectCmDeviceResultSIP>
```

```
   <StateInfo xsi:type="xsd:string">&lt;StateInfo&gt;&lt;Node Name=&quot;node70&quot;
SubsystemStartTime=&quot;1110841841&quot; StateId=&quot;8&quot;
TotalItemsFound=&quot;4&quot; TotalItemsReturned=&quot;4&quot;/&gt;&lt;Node
Name=&quot;node71&quot; SubsystemStartTime=&quot;1110688669&quot; StateId=&quot;5772&quot;
TotalItemsFound=&quot;0&quot;
TotalItemsReturned=&quot;0&quot;/&gt;&lt;/StateInfo&gt;</StateInfo>
  </ns1:SelectCmDeviceSIPResponse>
 </soapenv:Body>
</soapenv:Envelope>
```


Interface to Get Server Names and Cluster Name

The interface to get cluster name getServiceParameter, interface to get configured servers in cluster listProcessNodeByService, and interface to get configured devices in cluster listDeviceByNameAndClass are defined as part of the AXL Configuration API WSDL file. Send your queries to API question mailer on these interfaces.

PerfmonPort SOAP Service

The PerfmonPort (Performance Information Port) service comprises several operations that allow clients to do the following perfmon-related tasks:

- Collect perfmon counter data.

  Serviceability XML APIs provide two ways to collect perfmon data: session-based and single-transaction.
- Get a list of all perfmon objects and counter names that are installed in a particular host.
- Get a list of the current instances of a perfmon object.
- Get textual description of a perfmon counter.

Table 4-5 provides a summary of the SOAP PerfmonPort service operations.

Table 4-5 SOAP PerfmonPort Service Operations

Operation	Description	Reference
perfmonOpenSession	Allows client programs to obtain a session handle from the Serviceability XML APIs	PerfmonPort Service: perfmonOpenSession Operation, page 4-72
perfmonAddCounter	Adds an array of counters to a session handle	PerfmonPort Service: perfmonAddCounter Operation, page 4-73
perfmonRemoveCounter	Removes an array of counters from a session handle	PerfmonPort Service: perfmonRemoveCounter Operation, page 4-75
perfmonCollectSessionData	Collects perfmon data for all counters that have been added to the query handle	PerfmonPort Service: perfmonCollectSessionData Operation, page 4-77
perfmonCloseSession	Closes the session handle that the PerfmonOpenSession retrieved	PerfmonPort Service: perfmonCloseSession Operation, page 4-79
perfmonListInstance	Returns a list of instances of a Perfmon object in a particular host	PerfmonPort Service: perfmonListInstance Operation, page 4-80
perfmonQueryCounterDescription	Returns the help text of a particular counter	PerfmonPort Service: perfmonQueryCounterDescription Operation, page 4-82
perfmonListCounter	Returns the list of Perfmon objects and counters in a particular host	PerfmonPort Service: perfmonListCounter Operation, page 4-83
perfmonCollectCounterData	returns the Perfmon data for all counters that belong to an object in a particular host	PerfmonPort Service: perfmonCollectCounterData Operation, page 4-85

For a description for Perfmon error codes, see the "SOAP Fault Error Codes" section on page 4-144.

PerfmonPort Service: perfmonOpenSession Operation

Client programs submit the perfmonOpenSession operation to get a session handle from the Serviceability XML APIs. The client needs a session handle to do the session-based perfmon counter data collection. The session handle represents a universally unique identifier that is used once, which guarantees that no duplicate handles exist. Serviceability XML APIs keep the opened handles in cache. If no activity occurs on a handle for 25 hours, the Serviceability XML API removes the handle and renders it invalid.

Percentage counters require two samples to determine the average between the sample.

In a session-based perfmon data collection, use the following related operations:

- perfmonOpenSession
- perfmonAddCounter
- perfmonRemoveCounter
- perfmonCollectSessionData
- PerfmonCloseSession

After a client gets a session handle, it normally proceeds to submit the PerfmonAddCounter operation and then follows with the PerfmonCollectSessionData operation. PerfmonCollectSessionData specifies the main operation that collects perfmon data for the clients. When the client no longer needs the session handle, it should submit PerfmonCloseSession, so the Serviceability XML APIs can remove the handle from cache. Clients can dynamically add new counters to the session handle and remove counters from it by using the perfmonRemoveCounter operation while the session handle is still open.

Request Format

The PerfmonOpenSession operation takes no parameter.

The following example shows the PerfmonOpenSession request:

Example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonOpenSession
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/"/>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

PerfmonOpenSession returns a single element that is named SessionHandle. Its type specifies SessionHandleType, which is derived from xsd:string, and it contains the guide for the session handle.

The following example shows the PerfmonOpenSession response:

Example

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonOpenSessionResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SessionHandle
xsi:type="ns1:SessionHandleType">378273ba-ea59-11dc-8000-000000000000</SessionHandle>
    </ns1:PerfmonOpenSessionResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

PerfmonPort Service: perfmonAddCounter Operation

The perfmonAddCounter operation adds an array of counters to a session handle.

Request Format

The perfmonAddCounter operation takes the following parameters:

- **SessionHandle**—The type is SessionHandleType, which is derived from xsd:string. It contains the session handle that the previous perfmonOpenSession operation previously opened.
- **ArrayOfCounter**—The type for this element is ArrayOfCounterType, which is an array of counter elements. Each Counter element contains the name of a counter to be added to the session handle.

The following fragments from Serviceability XML APIs describe the types that this request uses:

```
    ...
<complexType name='ArrayOfCounterType'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
            <sequence>
                <element name='Counter'
                    type='tns:CounterType' minOccurs='1' maxOccurs='unbounded'/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
```

Note ArrayOfCounterType expects at least one Counter element in the array.

```
...
<complexType name='CounterType'>
    <sequence>
        <element name='Name' type='tns:CounterNameType'/>
    </sequence>
</complexType>
```

CounterType represents a structure, and it has a single element member: Name.

```
...
<simpleType name='CounterNameType'>
     <restriction base='string'/>
</simpleType>
```

The Name element that is of string-derived type contains the name of the counter.

The following example shows the perfmonAddCounter request with two counters. This example uses a single-reference accessor.

Example

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:tns="http://tempuri.org/"
    xmlns:types="http://tempuri.org/encodedTypes"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <q1:PerfmonAddCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <SessionHandle xsi:type="xsd:string">
                {1A490F1E-D82C-403F-9CF0-C4D4ABD6FF3E}
            </SessionHandle>
            <ArrayOfCounter soapenc:arrayType="q1:CounterType[2]">
                <Counter>
                    <Name>\\nozomi\process(inetinfo)\handle count</Name>
                </Counter>
                <Counter>
                    <Name>\\nozomi\process(csrss)\handle count</Name>
                </Counter>
            </ArrayOfCounter>
</q1:PerfmonAddCounter">
    </soap:Body>
</soap:Envelope
```

The following shows an example of the perfmonAddCounter request with three counters in the ArrayOfCounter parameter. This example uses multireference accessors.

Example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonAddCounter
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SessionHandle
xsi:type="ns1:SessionHandleType">38d47c54-ea59-11dc-8000-000000000000</SessionHandle>
      <ArrayOfCounter soapenc:arrayType="ns1:CounterType[2]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:CounterType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process\Nice</Name>
        </item>
        <item xsi:type="ns1:CounterType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process\PID</Name>
        </item>
      </ArrayOfCounter>
    </ns1:PerfmonAddCounter>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The following example shows that the perfmonAddCounter returns no output:

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonAddCounterResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/"/>
  </soapenv:Body>
</soapenv:Envelope>
```

If pefmonAddCounter fails to add one or more counters that are specified in the request, Serviceability XML APIs reply with a fault response. Some counters that are specified in the request may get successfully added, while others failed to be added.

In this case, Serviceability XML APIs reply with a fault. The Params element of CallInfo element specifies each failed counter name. Client programs can conclude that counter names, which are specified in the request but do not appear in the fault message, actually get added successfully to the query handle.

PerfmonPort Service: perfmonRemoveCounter Operation

The perfmonRemoveCounter operation removes an array of counters from a session handle.

Request Format

The perfmonRemoveCounter operation takes the following parameters:

- **SessionHandle**—The type is SessionHandleType which is derived from xsd:string. It contains the session handle that the PerfmonOpenSession operation opened previously.
- **ArrayOfCounter**—The type for this element is ArrayOfCounterType, which is an array of Counter elements. Each Counter element contain the name of a counter to be added to the session handle.

The following example shows a perfmonRemoveCounter request with three counters in the ArrayOfCounter parameter. This example uses single-reference accessor style.

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:tns="http://tempuri.org/"
    xmlns:types="http://tempuri.org/encodedTypes"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <q1:PerfmonRemoveCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <SessionHandle xsi:type="xsd:string">
                {1A490F1E-D82C-403F-9CF0-C4D4ABD6FF3E}
            </SessionHandle>
            <ArrayOfCounter soapenc:arrayType="q1:CounterType[2]">
                <Counter>
                    <Name>\\nozomi\process(inetinfo)\handle count</Name>
                </Counter>
                <Counter>
                    <Name>\\nozomi\process(csrss)\handle count</Name>
                </Counter>
                <Counter>
                    <Name>\\nozomi\process(regsvc)\handle count</Name>
```

```
            </Counter>
          </ArrayOfCounter>
</q1:PerfmonRemoveCounter">
    </soap:Body>
</soap:Envelope
```

The following example shows a perfmonRemoveCounter that uses multireference accessors.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonRemoveCounter
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SessionHandle
xsi:type="ns1:SessionHandleType">38d47c54-ea59-11dc-8000-000000000000</SessionHandle>
      <ArrayOfCounter soapenc:arrayType="ns1:CounterType[2]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:CounterType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process\Nice</Name>
        </item>
        <item xsi:type="ns1:CounterType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process\PID</Name>
        </item>
      </ArrayOfCounter>
    </ns1:PerfmonRemoveCounter>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The perfmonRemoveCounter operation returns no data in the response as shown by the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonRemoveCounterResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/"/>
  </soapenv:Body>
</soapenv:Envelope>
```

If the PefmonRemoveCounter operation fails to remove one or more counters that the request specifies, the Serviceability XML API replies with a fault response with semantics similar to PerfmonAddCounter. If some of the counters that are specified in the request get removed successfully, while others failed to be removed, the Serviceability XML API replies with a fault. The Params element of CallInfo element specifies each failed counter name. Client programs can conclude that counter names, which are specified in the request but do not appear in the fault message, actually get removed successfully from the query handle.

PerfmonPort Service: perfmonCollectSessionData Operation

The perfmonCollectSessionData operation collects the perfmon data for all counters that have been added to the query handle.

Request Format

The perfmonCollectSessionData operation takes the SessionHandle parameter. The type is SessionHandleType, which is derived from xsd:string. It contains the session handle that the perfmonOpenSession operation opened previously.

The following example shows a perfmonCollectSessionData request:

Example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonCollectSessionData
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SessionHandle
xsi:type="ns1:SessionHandleType">3accd2f4-ea59-11dc-8000-000000000000</SessionHandle>
    </ns1:PerfmonCollectSessionData>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The perfmonCollectSessionData operation returns the ArrayOfCounterInfo element that contains the value and status of all counters that were previously added to the session handle. The type for ArrayOfCounterInfo element specifies ArrayOfCounterInfoType, which is an array of CounterInfo elements.

The following fragments from Serviceability XML APIs .WSDL show the types that this response uses:

```
...
<complexType name='ArrayOfCounterInfoType'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
            <sequence>
                <element name='CounterInfo'
                    type='tns:CounterInfoType' minOccurs='1' maxOccurs='unbounded'/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
```

ArrayOfCounterInfoType has one or more CounterInfo elements in the array. The CounterInfo element includes the following type:

```
...
<complexType name='CounterInfoType'>
    <sequence>
        <element name='Name' type='tns:CounterNameType'/>
        <element name='Value' type='xsd:long'/>
        <element name='CStatus' type='xsd:unsignedInt'/>
    </sequence>
```

```
</complexType>
...
<simpleType name='CounterNameType'>
    <restriction base='string'/>
</simpleType>
```

CounterInfoType specifies a structure with the following element members.

- Name—A CounterNameType, derived from xsd:string, that contains the name of the counter that was previously added to the session handle.
- Value—A 64-bit signed integer (xsd:long) that contains the value of the counter.
- CStatus—Indicates whether the value of the counter was successfully retrieved. The type specifies a 32-bit unsigned integer (xsd:unsignedInt). First, check for the value of CStatus element before reading the Value element. If the value of CStatus equals 0 or 1, the Value element contains a good counter value. Otherwise, it indicates a failure in retrieving the counter value; ignore the Value element.

The following example shows a perfmonCollectSessionData response:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonCollectSessionDataResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ArrayOfCounterInfo soapenc:arrayType="ns1:CounterInfoType[6]"
xsi:type="soapenc:Array" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(0)\Nice
Percentage</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(0)\System
Percentage</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(0)\User
Percentage</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(_Total)\Nice
Percentage</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(_Total)\System
Percentage</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Processor(_Total)\User
Percentage</Name>
```

```
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">0</CStatus>
        </item>
      </ArrayOfCounterInfo>
    </ns1:PerfmonCollectSessionDataResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

PerfmonPort Service: perfmonCloseSession Operation

The perfmonCloseSession operation closes the session handle that the PerfmonOpenSession previously retrieved.

Request Format

The perfmonCloseSession operation takes the SessionHandle parameter. The type is SessionHandleType, which is derived from xsd:string. It contains the session handle that the perfmonOpenSession operation previously opened.

The following example shows a perfmonCloseSession request:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonCloseSession
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <SessionHandle
xsi:type="ns1:SessionHandleType">378273ba-ea59-11dc-8000-000000000000</SessionHandle>
    </ns1:PerfmonCloseSession>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The following example shows that the perfmonCloseSession does not return data in the response:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonCloseSessionResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/"/>
  </soapenv:Body>
</soapenv:Envelope>
```

PerfmonPort Service: perfmonListInstance Operation

The perfmonListInstance operation returns a list of instances of a perfmon object in a particular host. Instances of an object can dynamically come and go, and this operation returns the most recent list.

Request Format

The perfmonListIntance operation takes the following parameters:

- **Host**—The type is xsd:string. The Host parameter contains the name or address of the target server on which the object resides.
- **Object**—The type is xsd:string. The Object parameter contains the name of the object.

The following example shows a perfmonListInstance request with “nozomi” as the host and “Process” as the object parameter.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonListInstance
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <Host xsi:type="xsd:string">10.77.31.15</Host>
      <Object xsi:type="ns1:ObjectNameType">Process</Object>
    </ns1:PerfmonListInstance>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The perfmonListInstance returns an element that named ArrayOfInstance. The type for this element specifies ArrayOfInstanceType, which is an array of Instance elements. The following fragments from Serviceability XML APIs .WSDL file explain the types that this response uses:

```
...
<complexType name='ArrayOfInstanceType'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
            <sequence>
                <element name='Instance'
                    type='tns:InstanceType' minOccurs='0' maxOccurs='unbounded'/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
```

Note ArrayOfInstanceType can have 0 (zero) Instance elements, in which case the requested object is not of a multi-instance object.

```
...
<complexType name='InstanceType'>
    <sequence>
        <element name='Name' type='tns:InstanceNameType'/>
    </sequence>
</complexType>
```

The type for Instance element specifies InstanceType. It represents a structure with a single-element member: Name.

```
...
<simpleType name='InstanceNameType'>
    <restriction base='string'/>
</simpleType>
```

The Name element, whose type is InstanceNameType, which is derived from xsd:string, contains the name of the instance of the requested object.

The following example shows the response to the request in the previous example:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonListInstanceResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ArrayOfInstance soapenc:arrayType="ns1:InstanceType[133]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">init</Name>
        </item>
        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">migration_0</Name>
        </item>
        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">ksoftirqd_0</Name>
        </item>

...

        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">CTLProvider</Name>
        </item>
        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">capf</Name>
        </item>
        <item xsi:type="ns1:InstanceType">
          <Name xsi:type="ns1:InstanceNameType">CCMDirSync</Name>
        </item>
      </ArrayOfInstance>
    </ns1:PerfmonListInstanceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

PerfmonPort Service: perfmonQueryCounterDescription Operation

The perfmonQueryCounterDescription operation returns the help text of a particular counter.

Request Format

The perfmonQueryCounterDescription operation takes the Counter parameter. The name of the counter. Type is CounterNameType, which is derived from xsd:string.

The following example shows the perfmonQueryCounterDescription request:

Example

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <q1:PerfmonQueryCounterDescription
            xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <Counter xsi:type="xsd:string">\\nozomi\Server\Files Open</Counter>
        </q1:PerfmonQueryCounterDescription>
    </soap:Body>
</soap:Envelope>
```

Response Format

The perfmonQueryCounterDescription operation returns an element that is named HelpText that is of the xsd:string type. It contains the help text of the requested counter.

The following example shows the response to the request in the previous example.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <m:PerfmonQueryCounterDescriptionResponse
            xmlns:m="http://schemas.cisco.com/ast/soap/">
            <HelpText>The number of files currently opened in the server. Indicates
current server
activity.
</HelpText>
        </m:PerfmonQueryCounterDescriptionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

PerfmonPort Service: perfmonListCounter Operation

The perfmonListCounter operation returns the list of Perfmon objects and counters in a particular host.

Request Format

The perfmonListCounter operation takes the Host parameter. The type is xsd:string. The Host parameter contains the name or address of the target server from which the client wants to get the counter information.

The following example shows a perfmonListCounter request:

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <q1:PerfmonListCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <Host xsi:type="xsd:string">nozomi</Host>
        </q1:PerfmonListCounter>
    </soap:Body>
</soap:Envelope>
```

Response Format

The perfmonListCounter operation returns information that describes the hierarchical structure of Perfmon objects and counters. The body entry includes an ArrayOfObjectInfo element. The following fragments from the Serviceability XML APIs WSDL file describe the types that this response uses:

```
   ...
<complexType name='ArrayOfObjectInfoType'>
    <complexContent>
        <restriction base='SOAP-ENC:Array'>
            <sequence>
                <element name='ObjectInfo'
                    type='tns:ObjectInfoType' minOccurs='1' maxOccurs='unbounded'/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
```

The ArrayOfObjectInfo element comprises an array of ObjectInfo elements that have the following type:

```
...
<complexType name='ObjectInfoType'>
    <sequence>
        <element name='Name' type='tns:ObjectNameType'/>
        <element name='MultiInstance' type='xsd:boolean'/>
        <element name='ArrayOfCounter' type='tns:ArrayOfCounterType'/>
    </sequence>
</complexType>

...
<simpleType name='ObjectNameType'>
    <restriction base='string'/>
</simpleType>
```

The Name element, whose type is derived from string, describes the name of the object. MultiInstance element indicates whether the object has more than one instance. The ArrayOfCounter element acts as a container for an array of Counter elements that have the following types:

```
...
<complexType name='CounterType'>
    <sequence>
        <element name='Name' type='tns:CounterNameType'/>
    </sequence>
</complexType>
<simpleType name='CounterNameType'>
    <restriction base='string'/>
</simpleType>
```

The Name element, whose type is derived from xsd:string, describes the name of the counter.

Example

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <m:PerfmonListCounterResponse xmlns:m="http://schemas.cisco.com/ast/soap/">
            <ArrayOfObjectInfo SOAP-ENC:arrayType="m:ObjectInfoType[]">
                <ObjectInfo>
                    <Name>.NET CLR Memory</Name>
                    <MultiInstance>true</MultiInstance>
                    <ArrayOfCounter SOAP-ENC:arrayType="m:CounterType[]">
                        <Counter>
                            <Name># Gen 0 Collections</Name>
                        </Counter>
                        <Counter>
                            <Name># Gen 1 Collections</Name>
                        </Counter>
                        ...
                    </ArrayOfCounter>
                </ObjectInfo>
                <ObjectInfo>
                    <Name>.NET CLR LocksAndThreads</Name>
                    <MultiInstance>true</MultiInstance>
                    <ArrayOfCounter SOAP-ENC:arrayType="m:CounterType[]">
                        <Counter>
                            <Name>Total # of Contentions</Name>
                        </Counter>
                        <Counter>
                            <Name>Contention Rate / sec</Name>
                        </Counter>
                        <Counter>
                            <Name>Current Queue Length</Name>
                        </Counter>
                        ...
                    </ArrayOfCounter>
                </ObjectInfo>
                ...
            </ArrayOfObjectInfo>
        </m:PerfmonListCounterResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

PerfmonPort Service: perfmonCollectCounterData Operation

The perfmonCollectCounterData operation returns the perfmon data for all counters that belong to an object in a particular host. Unlike the session-based perfmon data collection, this operation collects all data in a single request/response transaction. If the object represents multiple-instance object, this operation always returns the most current instances of the object.

Request Format

The perfmonCollectCounterData operation takes the following parameters:

- **Host**—The type is xsd:string. It contains the address of the target server from which the client wants to get the counter information.
- **Object**—The type is ObjectNameType, which is derived from xsd:string. It contains the name of the perfmon object.

The following example shows a perfmonCollectCounterData request:

Example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:PerfmonCollectCounterData
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <Host xsi:type="xsd:string">10.77.31.15</Host>
      <Object xsi:type="ns1:ObjectNameType"></Object>
    </ns1:PerfmonCollectCounterData>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The perfmonCollectCounterData operation returns an ArrayOfCounterInfo element, which is an array of CounterInfo elements. CounterInfoType specifies a structure with the following three element members.

- **Name**—A CounterNameType, derived from xsd:string, that contains the name of the counter that was previously added to the session handle.
- **Value**—A 64-bit signed integer (xsd:long) that contains the value of the counter.
- **CStatus**—Indicates whether the value of the counter was successfully retrieved. The type specifies a 32-bit unsigned integer (xsd:unsignedInt). First, check for the value of CStatus element before reading the Value element. If the value of CStatus equals to 0 or 1, the Value element contains a good counter value. Otherwise, it indicates a failure in retrieving the counter value; ignore the Value element.

The following example shows a perfmonCollectCounterData response:

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <soapenv:Body>
    <ns1:PerfmonCollectCounterDataResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ArrayOfCounterInfo soapenc:arrayType="ns1:CounterInfoType[1995]"
xsi:type="soapenc:Array" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\% CPU
Time</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\% Memory
Usage</Name>
          <Value xsi:type="xsd:long">2</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\Data
Stack Size</Name>
          <Value xsi:type="xsd:long">344019</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name
xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\Nice</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name
xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\PID</Name>
          <Value xsi:type="xsd:long">9389</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(CCMDirSync)\Page
Fault Count</Name>
          <Value xsi:type="xsd:long">1</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>

...

<item xsi:type="ns1:CounterInfoType">
         <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(udevd)\Total CPU Time
Used</Name>
          <Value xsi:type="xsd:long">4</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(udevd)\UTime</Name>
          <Value xsi:type="xsd:long">0</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(udevd)\VmData</Name>
          <Value xsi:type="xsd:long">240</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(udevd)\VmRSS</Name>
          <Value xsi:type="xsd:long">692</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
```

```
        </item>
        <item xsi:type="ns1:CounterInfoType">
          <Name xsi:type="ns1:CounterNameType">\\10.77.31.15\Process(udevd)\VmSize</Name>
          <Value xsi:type="xsd:long">3020</Value>
          <CStatus xsi:type="xsd:unsignedInt">1</CStatus>
        </item>
      </ArrayOfCounterInfo>
    </ns1:PerfmonCollectCounterDataResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

ControlCenterServicesPort SOAP Service

The ControlCenterServicesPort service allows you to do Service Deploy, Service UnDeploy, Get Service List, and perform service starts and stops.

All ControlCenterServicesPort operations work only on the local node, which is specified by the NodeName element.

Table 4-6 provides a summary of the SOAP ControlCenterServicesPort service operations.

Table 4-6 SOAP ControlCenterServicesPort Service Operations

Operation	Description	Reference
soapGetStaticServiceList	Allows clients to perform a query for all services static specifications in Unified CM	ControlCenterServicesPort service: soapGetStaticServiceList Operation, page 4-88
soapGetServiceStatus	Allows clients to perform a list of deployable and undeployable services status query	ControlCenterServicesPort service: soapGetServiceStatus Operation, page 4-91
soapDoServiceDeployment	Allows clients to deploy or undeploy a list of services	ControlCenterServicesPort service: soapDoServiceDeployment Operation, page 4-101
soapDoControlServices	Allows clients to start or stop a list of service	ControlCenterServicesPort service: soapDoControlServices Operation, page 4-106
getProductInformationList	Provides information about the products that are installed on a given server	ControlCenterServicesPort service: getProductInformationList Operation, page 4-110

ControlCenterServicesPort service: soapGetStaticServiceList Operation

The soapGetStaticServiceList operation allows clients to perform a query for all services static specifications in Unified CM. It returns the array of service static specification, which is composed of service name, type (servlet or service), deployable or undeployable service, group name, and dependent services.

Request Format

The HTTP header should have following SOAP action and envelop information:

```
<operation name="soapGetStaticServiceList">
<soap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#ControlCenterServices#soapGetService
List"/>
    <input>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
</operation>
```

The **soapGetStaticServiceList** operation takes only one dummy string typed parameter.

Response Format

The response contains the ArrayOfServiceSpecification information that is defined as follows:

```
<complexType name="StaticServiceList">
    <sequence>
        <element name="Services" type="tns:ArrayOfServiceSpecification"/>
    </sequence>
</complexType>
```

ArrayOfServiceSpecification is an array of ServiceSpecification defined as follows:

```
<complexType name="ArrayOfServiceSpecification">
    <complexContent>
        <restriction base="SOAP-ENC:Array">
        <attribute ref="SOAP-ENC:arrayType"
              wsdl:arrayType="tns:ServiceSpecification[]"/>
        </restriction>
    </complexContent>
</complexType>
```

ServiceSpecification contains information defined as follows:

```
<complexType name="ServiceSpecification">
    <sequence>
        <element name="ServiceName" type="xsd:string"/>
        <element name="ServiceType" type="tns:ServiceTypes"/>
        <element name="Deployable" type="boolean"/>
        <element name="GroupName" type="xsd:string"/>
        <element name="DependentServices" type="tns:ArrayOfServices"/>
    </sequence>
</complexType>
```

where ServiceTypes defines the type of service, defined as follows:

```
<simpleType name="ServiceTypes">
    <restriction base="xsd:string">
        <enumeration value="Service"/>
        <enumeration value="Servlet"/>
    </restriction>
</simpleType>

ArrayOfServices defines the dependent services for the service, defined as an array:
<complexType name="ArrayOfServices">
    <complexContent>
        <restriction base="SOAP-ENC:Array">
            <attribute ref="SOAP-ENC:arrayType" wsdl:arrayType="xsd:string[]"/>
        </restriction>
    </complexContent>
</complexType>
```

Fault

Invalid Service Configuration Files

Static service specification comes from two service configuration xml files:

- /usr/local/cm/conf/ccmservice/ActivateConf.xml
- /usr/local/cm/conf/ccmservice/ServicesConf.xml

If one of the files does not exist or has an invalid format, an empty list of static service specification will appear in the response. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetStaticServiceListResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <StaticServiceList xsi:type="ns2:StaticServiceList"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <Services xsi:type="soapenc:Array" soapenc:arrayType="ns2:ServiceSpecification[0]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"/>
   </StaticServiceList>
  </ns1:GetStaticServiceListResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Request Example

The following example shows a request for getting a static service specification list:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:soapGetStaticServiceList
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse xsi:type="xsd:string">test</ServiceInformationResponse>
    </ns1:soapGetStaticServiceList>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example

The following example shows a response for getting a static service specification list:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:soapGetStaticServiceListResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <StaticServiceList xsi:type="ns2:StaticServiceList"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
        <Services soapenc:arrayType="ns2:ServiceSpecification[58]"
xsi:type="soapenc:Array" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns2:ServiceSpecification">

            <ServiceName xsi:type="xsd:string">Cisco AXL Web Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Database and Admin Services</GroupName>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
```

```
          <item xsi:type="ns2:ServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Serviceability Reporter</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring
Services</GroupName>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>

...

</Services>
      </StaticServiceList>
    </ns1:soapGetStaticServiceListResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

ControlCenterServicesPort service: soapGetServiceStatus Operation

The soapGetServiceStatus operation allows clients to perform a list of deployable and undeployable services status query. It returns the service status response information for the services that have been queried. It also allows getting all of the services current status by providing an empty list of services.

Request Format

The HTTP header should have following SOAP action and envelop information:

```
<operation name="soapGetServiceStatus">
    <soap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#ControlCenterServices#soapGetService
Status"/>
    <input>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
</operation>
```

The soapGetServiceStatus operation takes one array of service names for which their statuses are queried.

```
<GetServiceStatusList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item>service name</item>
</GetServiceStatusList>
```

If the array of service names is empty in the request, the response will contain service status information for all services in the system.

Response Format

The response contains the performed query information that is defined as follows:

```
<complexType name="ServiceInformationResponse">
    <sequence>
```

```
        <element name="ReturnCode" type="tns:ReturnCode"/>
        <element name="ReasonCode" type="xsd:integer"/>
        <element name="ReasonString" type="xsd:string"/>
        <element name="ServiceInfoList" type="tns:ArrayOfServiceInformation"/>
    </sequence>
</complexType>
```

where ReturnCode is a string format of return code:

```
<simpleType name="ReturnCode">
    <restriction base="xsd:string"/>
</simpleType>
```

ArrayOfServiceInformation is an array of ServiceInformation that defines service name, service status, reason code, reason string, service start time, and service up time.

```
<complexType name="ArrayOfServiceInformation">
    <complexContent>
    <restriction base="SOAP-ENC:Array">
        <attribute ref="SOAP-ENC:arrayType" wsdl:arrayType="tns:ServiceInformation[ ]"/>
    </restriction>
</complexContent>
</complexType>
<complexType name="ServiceInformation">
    <sequence>
        <element name="ServiceName" type="xsd:string"/>
        <element name="ServiceStatus" type="tns:ServiceStatus"/>
        <element name="ReasonCode" type="xsd:integer"/>
        <element name="ReasonCodeString" type="xsd:string"/>
        <element name="StartTime" type="xsd:string"/>
        <element name="UpTime" type="xsd:integer"/>
    </sequence>
</complexType>
```

ServiceStatus defines the enumerated status for the service, as follows:

```
<simpleType name="ServiceStatus">
    <restriction base="xsd:string">
        <enumeration value="Started"/>
        <enumeration value="Stopped"/>
        <enumeration value="Starting"/>
        <enumeration value="Stopping"/>
        <enumeration value="Unknown"/>
    </restriction>
</simpleType>
```

The following details apply about ServiceStatus, ReasonCode, and ReasonCodeString.

- ServiceStatus is either Started or Stopped. If Started, StartTime gives the time that the service is started in a time string format; UpTime gives the time in seconds since the service was started.
- The ReasonCode and ReasonCodeString explain the reason that the service is Stopped:
  - If a deployable service is activated and stopped by an administrator, its status is Stopped, the ReasonCode equals -1019, and the corresponding ReasonCodeString specifies “Component is not running”.
  - If a deployable service is deactivated, its status is Stopped, the ReasonCode equals -1068, and the corresponding ReasonCodeString specifies “Service Not Activated”.
  - If a nondeployable item is stopped by an administrator, its status could be Stopped with ReasonCode -1019, and the corresponding ReasonCodeString “Component is not running.”

    The complete listing of ReasonCode and ReasonCodeString follows:

    ```
    -1000: "Component already initialized"
    ```

```
-1001: "Entry replaced"
-1002: "Component not initialized"
-1003: "Component is running"
-1004: "Entry not found"
-1005: "Unable to process event"
-1006: "Registration already present"
-1007: "Unsuccessful completion"
-1008: "Registration not found"
-1009: "Missing or invalid environment variable"
-1010: "No such service"
-1011: "Component is reserved for platform"
-1012: "Bad arguments"
-1013: "Internal error"
-1014: "Entry was already present"
-1015: "Error opening IPC"
-1016: "No license available"
-1017: "Error opening file"
-1018: "Error reading file"
-1019: "Component is not running"
-1020: "Signal ignored"
-1021: "Notification ignored"
-1022: "Buffer overflow"
-1023: "Cannot parse record or entry"
-1024: "Out of memory"
-1025: "Not connected"
-1026: "Component already exists"
-1027: "Message was truncated"
-1028: "Component is empty"
-1029: "Operation is pending"
-1030: "Transaction does not exist"
-1031: "Operation timed-out"
-1032: "File is locked"
-1033: "Feature is not implemented yet"
-1034: "Alarm was already set"
-1035: "Alarm was already clear"
-1036: "Dependency is in active state"
-1037: "Dependency is not in active state"
-1038: "Circular dependencies detected"
-1039: "Component already started"
-1040: "Component already stopped"
-1041: "Dependencies still pending"
-1042: "Requested process state transition not allowed"
-1043: "No changes"
-1044: "Boundary violation for data structure"
-1045: "Operation not supported"
-1046: "Process recovery in progress"
-1047: "Operation pending on scheduled restart"
-1048: "Operation pending on active dependencies"
-1049: "Operation pending on active dependents"
-1050: "Shutdown is in progress"
-1051: "Invalid Table Handle"
-1052: "Data Base not initialized"
-1053: "Data Directory"
-1054: "Table Full"
-1055: "Deleted Data"
-1056: "No Such Record"
-1057: "Component already in specified state"
-1058: "Out of range"
-1059: "Cannot create object"
-1060: "MSO refused, standby system not ready."
-1061: "MSO refused, standby state update still in progress. Try again later."
-1062: "MSO refused, standby state update failed.
       Verify configuration on standby."
-1063: "MSO refused, Warm start-up in progress."
```

```
-1064: "MSO refused, Warm start-up Failed."
-1065: "MSO refused, System is not in active state"
-1066: "MSO refused, Detected standalone Flag"
-1067: "Invalid Token presented in reques"
-1068: "Service Not Activated"
-1069: "Commanded Out of Service"
-1070: "Multiple Modules have error"
-1071: "Encountered exception"
-1072: "Invalid context path was specified"
-1073: "No context exists"
-1074: "No context path was specified"
-1075: "Application already exists"
```

Fault

Invalid Service: Non-existing Service

If the request for getting the service status is for an invalid service name, such as a nonexistent service name, the ReasonCode -1010 and the ReasonCodeString “No such service” will appear in the response. The following request and response example applies for getting service status for “Cisco WrongService.”

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetServiceStatus soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <GetServiceStatusList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
    <item>Cisco WrongService</item>
   </GetServiceStatusList>
  </ns1:GetServiceStatus>
 </soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetServiceStatusResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1010</ReasonCode>
    <ReasonString xsi:type="xsd:string">No such service</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:GetServiceStatusResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Request Example Cisco Tftp Service Status

The following request example for getting “Cisco Tftp” service status applies:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetServiceStatus
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <GetServiceStatusList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
        <item>Cisco Tftp</item>
      </GetServiceStatusList>
    </ns1:GetServiceStatus>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example for Cisco Tftp Service Status

The following response example for getting “Cisco Tftp” service status applies:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetServiceStatusResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
        <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonString xsi:type="xsd:string" xsi:nil="true"/>
        <ServiceInfoList xsi:type="soapenc:Array"
soapenc:arrayType="ns2:ServiceInformation[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item>
            <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
            <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
            <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
            <ReasonCodeString xsi:type="xsd:string">
            </ReasonCodeString>
            <StartTime xsi:type="xsd:string">Tue Sep 14 14:29:43 2004</StartTime>
            <UpTime xsi:type="xsd:integer">11800</UpTime>
          </item>
        </ServiceInfoList>
      </ServiceInformationResponse>
    </ns1:GetServiceStatusResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Request Example for Empty Array of Service Names

The following request example for getting service status when providing an empty array of service names applies:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
 <soapenv:Body>
  <ns1:GetServiceStatus soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <GetServiceStatusList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[0]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"/>
  </ns1:GetServiceStatus>
 </soapenv:Body>
</soapenv:Envelope>
```

Response Example for Empty Array of Service Names

The response for the preceding request gets the current status for all services as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetServiceStatusResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
    <ReasonString xsi:type="xsd:string" xsi:nil="true"/>
    <ServiceInfoList xsi:type="soapenc:Array"
soapenc:arrayType="ns2:ServiceInformation[43]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>
      <ServiceName xsi:type="xsd:string">A Red Hat DB</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:14 2004</StartTime>
      <UpTime xsi:type="xsd:integer">186704</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco AMC Service</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Mon Dec  6 17:51:26 2004</StartTime>
      <UpTime xsi:type="xsd:integer">161372</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco AXL Web Service</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:04 2004</StartTime>
      <UpTime xsi:type="xsd:integer">73674</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">CiscoUnified CM SNMP Service</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:19 2004</StartTime>
      <UpTime xsi:type="xsd:integer">186699</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco CDP</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:19 2004</StartTime>
```

```
        <UpTime xsi:type="xsd:integer">186699</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Cisco CDP Agent</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:19 2004</StartTime>
        <UpTime xsi:type="xsd:integer">186699</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Cisco CDR Agent</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 10:50:38 2004</StartTime>
        <UpTime xsi:type="xsd:integer">186620</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Cisco CTIManager</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
        <UpTime xsi:type="xsd:integer">186698</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Cisco CTL Provider</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
        <UpTime xsi:type="xsd:integer">186698</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Unified CM</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 12:43:07 2004</StartTime>
        <UpTime xsi:type="xsd:integer">179871</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Unified CM Admin</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:01 2004</StartTime>
        <UpTime xsi:type="xsd:integer">73677</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Cisco Unified Communications Manager Attendant
Console
             Server</ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
        <UpTime xsi:type="xsd:integer">186698</UpTime>
       </item>
       <item>
        <ServiceName xsi:type="xsd:string">Unified CM Personal Directory
          </ServiceName>
        <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
        <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:02 2004</StartTime>
        <UpTime xsi:type="xsd:integer">73676</UpTime>
       </item>
       <item>
```

```
 <ServiceName xsi:type="xsd:string">Unified CM Serviceability<
      /ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:13 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73665</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Unified CM Serviceability RTMT
  </ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:03 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73675</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco DRF Local</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1019</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string">Component is not running</ReasonCodeString>
 <StartTime xsi:type="xsd:string" xsi:nil="true"/>
 <UpTime xsi:type="xsd:integer">-1</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Database Layer Monitor</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186698</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco DirSync</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186698</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Electronic Notification</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186698</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Extended Functions</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186698</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Extension Mobility</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:57 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73681</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Extension Mobility Application
  </ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
```

```
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:39 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73699</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco IP Voice Media Streaming App</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 12:30:51 2004</StartTime>
 <UpTime xsi:type="xsd:integer">180607</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Log Partition Monitoring Tool</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186698</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Messaging Interface</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1019</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string">Component is not running</ReasonCodeString>
 <StartTime xsi:type="xsd:string" xsi:nil="true"/>
 <UpTime xsi:type="xsd:integer">-1</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco RIS Data Collector</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 16:25:25 2004</StartTime>
 <UpTime xsi:type="xsd:integer">166533</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco RTMT Reporter Servlet</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:56 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73682</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco SOAP -Log Collection APIs</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:56 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73682</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco SOAP - Performance Monitoring APIs
   </ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:58 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73680</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco SOAP -Real-Time Service APIs</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:59 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73679</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Serviceability Reporter</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
```

```
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:21 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186697</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Syslog Agent</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:21 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186697</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 17:51:20 2004</StartTime>
 <UpTime xsi:type="xsd:integer">161378</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco Tomcat</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:12:35 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73703</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco WebDialer Web Service</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Tue Dec  7 18:13:00 2004</StartTime>
 <UpTime xsi:type="xsd:integer">73678</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Host Resources Agent</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:56 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186662</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">MIB2 Agent</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:58 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186660</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Native Agent Adaptor</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
 <StartTime xsi:type="xsd:string">Mon Dec  6 10:49:59 2004</StartTime>
 <UpTime xsi:type="xsd:integer">186659</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">SNMP Master Agent</ServiceName>
 <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
 <ReasonCode xsi:type="xsd:integer">-1019</ReasonCode>
 <ReasonCodeString xsi:type="xsd:string">Component is not running</ReasonCodeString>
 <StartTime xsi:type="xsd:string" xsi:nil="true"/>
 <UpTime xsi:type="xsd:integer">-1</UpTime>
</item>
<item>
 <ServiceName xsi:type="xsd:string">Cisco CAR Scheduler</ServiceName>
```

```
      <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1068</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string">Service Not Activated </ReasonCodeString>
      <StartTime xsi:type="xsd:string" xsi:nil="true"/>
      <UpTime xsi:type="xsd:integer">-1</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco CAR Web Service</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1068</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string">Service Not Activated </ReasonCodeString>
      <StartTime xsi:type="xsd:string" xsi:nil="true"/>
      <UpTime xsi:type="xsd:integer">-1</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco CDR Repository Manager</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1068</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string">Service Not Activated </ReasonCodeString>
      <StartTime xsi:type="xsd:string" xsi:nil="true"/>
      <UpTime xsi:type="xsd:integer">-1</UpTime>
     </item>
     <item>
      <ServiceName xsi:type="xsd:string">Cisco SOAP - CDRonDemand Service</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Stopped</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1068</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string">Service Not Activated </ReasonCodeString>
      <StartTime xsi:type="xsd:string" xsi:nil="true"/>
      <UpTime xsi:type="xsd:integer">-1</UpTime>
     </item>
    </ServiceInfoList>
   </ServiceInformationResponse>
  </ns1:GetServiceStatusResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

ControlCenterServicesPort service: soapDoServiceDeployment Operation

The soapDoServiceDeployment operation allows clients to deploy or undeploy a list of services. It returns the services response information for the services that are requested to be deployed or undeployed. You can use this API only to deploy or undeploy a deployable service, a service with the Deployable attribute set to True in the response from getting the static service specification. The API does not allow clients to deploy or undeploy an empty list of services.

This API only activates services the node that is specified by the NodeName element. NodeName must specify the local node.

Request Format

The HTTP header should have following SOAP action and envelop information:

```
<operation name="soapDoServiceDeployment">
    <soap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#ControlCenterServices#soapDoServiceD
eployment"/>
    <input>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
```

```
</operation>
```

The soapDoServiceDeployment operation takes one array of service names for which their services are either deployed or undeployed:

```
<soapenv:Body>
    <ns1:DoServiceDeployment
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <DeploymentServiceRequest href="#id0"/>
    </ns1:DoServiceDeployment>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:DeploymentServiceRequest"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
      <NodeName xsi:type="xsd:string">172.19.240.61</NodeName>
      <DeployType href="#id1"/>
      <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
        <item>service name</item>
      </ServiceList>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:DeployType"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Deploy</multiRef>
  </soapenv:Body>
```

Response Format

The response contains the performed query information as defined in the previous "Response Format" section on page 4-91.

Fault

Invalid Service: Nonexisting Service

If the request to activate or deactivate a service is for an invalid service name, such as a nonexistent service name, the ReasonCode -1010 and the ReasonCodeString "No such service" will appear in the response.

The following request and response example applies for activating the service "Cisco WrongService."

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeployment
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <DeploymentServiceRequest xsi:type="ns2:DeploymentServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string">172.19.240.99</NodeName>
    <DeployType xsi:type="ns2:DeployType">Deploy</DeployType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>Cisco WrongService</item>
    </ServiceList>
   </DeploymentServiceRequest>
```

```
  </ns1:DoServiceDeployment>
 </soapenv:Body>
</soapenv:Envelope>


<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeploymentResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1010</ReasonCode>
    <ReasonString xsi:type="xsd:string">No such service</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoServiceDeploymentResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Invalid Service: Nondeployable Service

If the request to activate or deactivate a service is for a nondeployable service, a service with the Deployable attribute set to False in the response for getting the static service specification, such as service “SNMP Master Agent,” the ReasonCode -1045 and the ReasonCodeString “Operation not supported” will appear in the response.

The following request and response example applies for activating the service “SNMP Master Agent.”

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeployment
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <DeploymentServiceRequest xsi:type="ns2:DeploymentServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string">172.19.240.99</NodeName>
    <DeployType xsi:type="ns2:DeployType">Deploy</DeployType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>SNMP Master Agent</item>
    </ServiceList>
   </DeploymentServiceRequest>
  </ns1:DoServiceDeployment>
 </soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeploymentResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
```

```
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1045</ReasonCode>
    <ReasonString xsi:type="xsd:string">Operation not supported</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoServiceDeploymentResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Invalid Service: Empty List of Services

If the request to activate or deactivate a service provides an empty list of services, the ReasonCode -1045 and the ReasonCodeString "Operation not supported" will appear in the response.

The following request and response example applies for activating the service without providing any service name:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeployment
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <DeploymentServiceRequest xsi:type="ns2:DeploymentServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string">172.19.240.99</NodeName>
    <DeployType xsi:type="ns2:DeployType">Deploy</DeployType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[0]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"/>
   </DeploymentServiceRequest>
  </ns1:DoServiceDeployment>
 </soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeploymentResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1045</ReasonCode>
    <ReasonString xsi:type="xsd:string">Operation not supported</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoServiceDeploymentResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Request Example

The request example for deploying "Cisco Tftp" service follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <soapenv:Body>
    <ns1:DoServiceDeployment
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <DeploymentServiceRequest href="#id0"/>
    </ns1:DoServiceDeployment>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:DeploymentServiceRequest"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
      <NodeName xsi:type="xsd:string">172.19.240.61</NodeName>
      <DeployType href="#id1"/>
      <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
        <item>Cisco Tftp</item>
      </ServiceList>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:DeployType"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Deploy</multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example

The response example for deploying “Cisco Tftp” service follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoServiceDeploymentResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
    <ReasonString xsi:type="xsd:string" xsi:nil="true"/>
    <ServiceInfoList xsi:type="soapenc:Array"
soapenc:arrayType="ns2:ServiceInformation[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>
      <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
      <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
      <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
      <ReasonCodeString xsi:type="xsd:string"> </ReasonCodeString>
      <StartTime xsi:type="xsd:string">Wed Sep 15 13:59:28 2004</StartTime>
      <UpTime xsi:type="xsd:integer">6</UpTime>
     </item>
    </ServiceInfoList>
   </ServiceInformationResponse>
  </ns1:DoServiceDeploymentResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

ControlCenterServicesPort service: soapDoControlServices Operation

The soapDoControlServices operation allows clients to start or stop a list of service. It returns the services response information for the services that are requested to get started or stopped. The API does not allow clients to stop the following non-stop services:

- A Cisco DB
- Cisco Tomcat

The API also does not allow clients to provide an empty list of services when trying to start or stop the services.

Request Format

HTTP header should have following SOAP action and envelop information:

```
<operation name="soapDoControlServices">
    <soap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#ControlCenterServices#soapDoCont
rolServices"/>
    <input>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
        <soap:body use="encoded" namespace="http://schemas.cisco.com/ast/soap/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
</operation>
```

The soapDoControlServices operation takes one array of service names for which their services are either started or stopped.

```
<multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ControlServiceRequest"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
      <NodeName xsi:type="xsd:string" xsi:nil="true"/>
      <ControlType href="#id1"/>
      <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
        <item>service name</item>
      </ServiceList>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:ControlType"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Start</multiRef>
```

Response Format

The response contains the performed query information as defined in the previous "Response Format" section on page 4-91.

Fault

Invalid Service: Nonexisting Service

If the request of start/stop service for an invalid service name, such as a nonexisting service name, the ReasonCode -1010 and the ReasonCodeString “No such service” will be the response. The following request and response example applies for starting the service “Cisco WrongService.”

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServices soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ControlServiceRequest xsi:type="ns2:ControlServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string" xsi:nil="true"/>
    <ControlType xsi:type="ns2:ControlType">Start</ControlType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>Cisco WrongService</item>
    </ServiceList>
   </ControlServiceRequest>
  </ns1:DoControlServices>
 </soapenv:Body>
</soapenv:Envelope>


<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServicesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1010</ReasonCode>
    <ReasonString xsi:type="xsd:string">No such service</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoControlServicesResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Invalid Service: Nonstop Service

If the request of stop service for a nonstop service, such as service “Cisco Tomcat”, the ReasonCode -1045 and the ReasonCodeString “Operation not supported” will be the response. The following request and response example applies for stopping the service “Cisco Tomcat.”

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServices soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ControlServiceRequest xsi:type="ns2:ControlServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string" xsi:nil="true"/>
```

```
    <ControlType xsi:type="ns2:ControlType">Stop</ControlType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>Cisco Tomcat</item>
    </ServiceList>
   </ControlServiceRequest>
  </ns1:DoControlServices>
 </soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServicesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1045</ReasonCode>
    <ReasonString xsi:type="xsd:string">Operation not supported</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoControlServicesResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Invalid Service: Empty List of Services

If the request of start or stop service is with an empty list of services, the ReasonCode -1045 and the ReasonCodeString “Operation not supported” will be the response. The following request and response example applies for stopping the service without providing any service name.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServices soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ControlServiceRequest xsi:type="ns2:ControlServiceRequest"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <NodeName xsi:type="xsd:string" xsi:nil="true"/>
    <ControlType xsi:type="ns2:ControlType">Stop</ControlType>
    <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[0]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"/>
   </ControlServiceRequest>
  </ns1:DoControlServices>
 </soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:DoControlServicesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
    <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
    <ReasonCode xsi:type="xsd:integer">-1045</ReasonCode>
```

```
    <ReasonString xsi:type="xsd:string">Operation not supported</ReasonString>
    <ServiceInfoList xsi:type="ns2:ServiceInformation" xsi:nil="true"/>
   </ServiceInformationResponse>
  </ns1:DoControlServicesResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Invalid Service: Service with Stopping Status

If the request is to stop a service, and the service status is Stopping, the ReasonCode -1045 and the ReasonCodeString "Operation not supported" will be the response. The request and response example appears very similar to the preceding example.

Request Example

The request example for starting "Cisco Tftp" service is:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:DoControlServices
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ControlServiceRequest href="#id0"/>
    </ns1:DoControlServices>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ControlServiceRequest"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
      <NodeName xsi:type="xsd:string" xsi:nil="true"/>
      <ControlType href="#id1"/>
      <ServiceList xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
        <item>Cisco Tftp</item>
      </ServiceList>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:ControlType"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Start</multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example

The response example for starting "Cisco Tftp" service follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:DoControlServicesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse xsi:type="ns2:ServiceInformationResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
        <ReturnCode xsi:type="ns2:ReturnCode">0</ReturnCode>
```

```
        <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
        <ReasonString xsi:type="xsd:string" xsi:nil="true"/>
        <ServiceInfoList xsi:type="soapenc:Array"
soapenc:arrayType="ns2:ServiceInformation[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item>
            <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
            <ServiceStatus xsi:type="ns2:ServiceStatus">Started</ServiceStatus>
            <ReasonCode xsi:type="xsd:integer">-1</ReasonCode>
            <ReasonCodeString xsi:type="xsd:string">
            </ReasonCodeString>
            <StartTime xsi:type="xsd:string">Tue Sep 14 19:36:08 2004</StartTime>
            <UpTime xsi:type="xsd:integer">6</UpTime>
          </item>
        </ServiceInfoList>
      </ServiceInformationResponse>
    </ns1:DoControlServicesResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

ControlCenterServicesPort service: getProductInformationList Operation

The getProductInformationList operation provides information about the products that are installed on a given server. This information includes:

- Active Server Version
- Primary Node name
- SecondaryNode name (if any)
- Array Of Installed Products

  Each installed product provides the following information:

  - ProductName
  - Product Version
  - Product Description
  - Product ID
  - Short Name for the product
- Array Of Product Service Specification

  Each Product Service Specification provides the following information:

  - Service Name
  - Service Type
  - Deployable value
  - GroupName
  - ProductID
  - Array of DependentServices (if any).

For each server in the cluster, clients are expected to send one request to get all this information. The system requires clients to send this request only once during initialization.

Request Format

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetProductInformationList
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse
xsi:type="xsd:string">getProduct</ServiceInformationResponse>
    </ns1:GetProductInformationList>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Format

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetProductInformationListResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <GetProductInformationListResponse xsi:type="ns2:GetProductInformationListResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
        <ActiveServerVersion xsi:type="xsd:string">6.0.0.9381-5</ActiveServerVersion>
        <PrimaryNode xsi:type="xsd:string">irv3-ccm4</PrimaryNode>
        <SecondaryNode xsi:type="xsd:string">
        </SecondaryNode>
        <Products soapenc:arrayType="ns2:InstalledProduct[3]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns2:InstalledProduct">
            <ProductName xsi:type="xsd:string">Cisco Unified CallManager</ProductName>
            <ProductVersion xsi:type="xsd:string">6.0.0.9381-5</ProductVersion>
            <ProductDescription xsi:type="xsd:string">CallManager temporary
description</ProductDescription>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <ShortName xsi:type="xsd:string">CCM</ShortName>
          </item>
          <item xsi:type="ns2:InstalledProduct">
            <ProductName xsi:type="xsd:string">Cisco Unity Connection</ProductName>
            <ProductVersion xsi:type="xsd:string">6.0.0.9381-5</ProductVersion>
            <ProductDescription xsi:type="xsd:string">Unity Connection temporary
description</ProductDescription>
            <ProductID xsi:type="xsd:string">UnityConnection</ProductID>
            <ShortName xsi:type="xsd:string">CUC</ShortName>
          </item>
          <item xsi:type="ns2:InstalledProduct">
            <ProductName xsi:type="xsd:string">Common Services</ProductName>
            <ProductVersion xsi:type="xsd:string">6.0.0.9381-5</ProductVersion>
            <ProductDescription xsi:type="xsd:string">Common Services for all
Products</ProductDescription>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <ShortName xsi:type="xsd:string">SYS</ShortName>
          </item>
        </Products>
        <Services soapenc:arrayType="ns2:ProductServiceSpecification[58]"
xsi:type="soapenc:Array" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
```

```
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
        ..
        ..
         </Services>
      </GetProductInformationListResponse>
    </ns1:GetProductInformationListResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Fault

The system issues a standard SOAP fault in the event of a failure.

Request Example

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetProductInformationList
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse xsi:type="xsd:string">test</ServiceInformationResponse>
    </ns1:GetProductInformationList>
</soapenv:Body>
</soapenv:Envelope>
```

Response Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:GetProductInformationListResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <GetProductInformationListResponse xsi:type="ns2:GetProductInformationListResponse"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/ControlCenterServices/">
        <ActiveServerVersion xsi:type="xsd:string">7.0.0.39700-8</ActiveServerVersion>
        <PrimaryNode xsi:type="xsd:string">CISCART15</PrimaryNode>
        <SecondaryNode xsi:type="xsd:string">
        </SecondaryNode>
        <Products soapenc:arrayType="ns2:InstalledProduct[2]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns2:InstalledProduct">
            <ProductName xsi:type="xsd:string">Cisco Unified Communications
Manager</ProductName>
```

```
            <ProductVersion xsi:type="xsd:string">7.0.0.39700-8</ProductVersion>
            <ProductDescription xsi:type="xsd:string">Stand Alone Cisco Unified
Communications Manager</ProductDescription>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <ShortName xsi:type="xsd:string">CCM</ShortName>
          </item>
          <item xsi:type="ns2:InstalledProduct">
            <ProductName xsi:type="xsd:string">Common Services</ProductName>
            <ProductVersion xsi:type="xsd:string">7.0.0.39700-8</ProductVersion>
            <ProductDescription xsi:type="xsd:string">Common Services for all
Products</ProductDescription>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <ShortName xsi:type="xsd:string">SYS</ShortName>
          </item>
        </Products>
        <Services soapenc:arrayType="ns2:ProductServiceSpecification[58]"
xsi:type="soapenc:Array" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco AXL Web Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Database and Admin Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Serviceability Reporter</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring
Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco DirSync</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Directory Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Messaging Interface</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
```

```
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Unified Mobile Voice Access
Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco IP Voice Media Streaming
App</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CTIManager</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices soapenc:arrayType="xsd:string[1]" xsi:type="soapenc:Array">
              <item xsi:type="xsd:string">Cisco CallManager</item>
            </DependentServices>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager Attendant Console
Server</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CTI Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Extension Mobility</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco IP Manager Assistant</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CTI Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco SOAP - CDRonDemand
Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CDR Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
```

```
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CTL Provider</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">Security Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Extended Functions</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">Voice Quality Reporter Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco WebDialer Web Service</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">CTI Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Certificate Authority Proxy
Function</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">Security Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CAR Web Service</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">CDR Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CallManager SNMP
Service</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">Performance and Monitoring
Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Bulk Provisioning
Service</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
              <GroupName xsi:type="xsd:string">Database and Admin Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Dialed Number Analyzer</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">true</Deployable>
```

```
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco DHCP Monitor Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices soapenc:arrayType="xsd:string[1]" xsi:type="soapenc:Array">
              <item xsi:type="xsd:string">Cisco Database Layer Monitor</item>
            </DependentServices>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco TAPS Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">Database and Admin Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager Serviceability
RTMT</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco DRF Master</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Backup and Restore Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager
Serviceability</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">System Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">A Cisco DB</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Platform Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">A Cisco DB Replicator</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Platform Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
```

```
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">Cisco Tomcat</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">SNMP Master Agent</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">Cisco Database Layer Monitor</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">DB Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">MIB2 Agent</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">Host Resources Agent</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">Native Agent Adapter</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">System Application Agent</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
<item xsi:type="ns2:ProductServiceSpecification">
  <ServiceName xsi:type="xsd:string">Cisco CDP Agent</ServiceName>
  <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
  <Deployable xsi:type="xsd:boolean">false</Deployable>
  <GroupName xsi:type="xsd:string">Platform Services</GroupName>
  <ProductID xsi:type="xsd:string">Common</ProductID>
  <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
</item>
```

```
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Syslog Agent</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">Platform Services</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco RTMT Reporter Servlet</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Log Partition Monitoring
Tool</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices soapenc:arrayType="xsd:string[1]" xsi:type="soapenc:Array">
                <item xsi:type="xsd:string">Cisco Database Layer Monitor</item>
              </DependentServices>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CDP</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">System Services</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">SOAP -Real-Time Service APIs</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">SOAP Services</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">SOAP -Performance Monitoring
APIs</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">SOAP Services</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">SOAP -Log Collection APIs</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">SOAP Services</GroupName>
              <ProductID xsi:type="xsd:string">Common</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco DRF Local</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
```

```
            <GroupName xsi:type="xsd:string">Backup and Restore Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Certificate Expiry
Monitor</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Platform Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Trace Collection
Servlet</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">System Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Trace Collection
Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">System Services</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco Tomcat Stats Servlet</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco RIS Data Collector</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco AMC Service</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">Performance and Monitoring</GroupName>
            <ProductID xsi:type="xsd:string">Common</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
          <item xsi:type="ns2:ProductServiceSpecification">
            <ServiceName xsi:type="xsd:string">Cisco CallManager Personal
Directory</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
            <Deployable xsi:type="xsd:boolean">false</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <ProductID xsi:type="xsd:string">CallManager</ProductID>
            <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
          </item>
```

```
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CDR Repository Manager</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">CDR Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices soapenc:arrayType="xsd:string[2]" xsi:type="soapenc:Array">
                <item xsi:type="xsd:string">Cisco Database Layer Monitor</item>
                <item xsi:type="xsd:string">A Cisco DB</item>
              </DependentServices>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CallManager Admin</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">Admin Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco License Manager</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">Platform Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CDR Agent</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">CDR Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices soapenc:arrayType="xsd:string[2]" xsi:type="soapenc:Array">
                <item xsi:type="xsd:string">Cisco Database Layer Monitor</item>
                <item xsi:type="xsd:string">A Cisco DB</item>
              </DependentServices>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco Extension Mobility
Application</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">CM Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CallManager Cisco IP Phone
Services</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">CM Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
            </item>
            <item xsi:type="ns2:ProductServiceSpecification">
              <ServiceName xsi:type="xsd:string">Cisco CAR Scheduler</ServiceName>
              <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
              <Deployable xsi:type="xsd:boolean">false</Deployable>
              <GroupName xsi:type="xsd:string">CDR Services</GroupName>
              <ProductID xsi:type="xsd:string">CallManager</ProductID>
              <DependentServices soapenc:arrayType="xsd:string[2]" xsi:type="soapenc:Array">
                <item xsi:type="xsd:string">Cisco Database Layer Monitor</item>
```

```
                <item xsi:type="xsd:string">A Cisco DB</item>
              </DependentServices>
            </item>
          </Services>
        </GetProductInformationListResponse>
      </ns1:GetProductInformationListResponse>
    </soapenv:Body>
</soapenv:Envelope>
```

LogCollectionPort SOAP Service

This section describes the operations for the LogCollectionPort service, which provides operations for searching for and collecting log files for a set of services.

Table 4-7 provides a summary of the SOAP LogCollectionPort service operations.

Table 4-7 SOAP LogCollectionPort Service Operations

Operation	Description	Reference
listNodeServiceLogs	Returns the location of their log files for each service	LogCollectionPort service: listNodeServiceLogs Operation, page 4-121
selectLogFiles	Takes FileSelectionCriteria object as an input parameter and returns the file names and location for that object	LogCollectionPort Service: selectLogFiles Operation, page 4-124

LogCollectionPort service: listNodeServiceLogs Operation

The listNodeServiceLogs operation returns the location of their log files for each service. This information includes the node names in the cluster and the lists of service names and system log names.

Request Format

The input is a dummy ListRequest Handle.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:ListNodeServiceLogs
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ListRequest href="#id0"/>
  </ns1:ListNodeServiceLogs>
  <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ListRequest" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/LogCollection/">handle</multiRef>
 </soapenv:Body>
</soapenv:Envelope>
```

Response Format

The response is an array of NodeServiceLogList.

```
message="impl:listNodeServiceLogsResponse" name="listNodeServiceLogsResponse" />
String name;
String[] serviceLog;
String[] systemlog;

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:ListNodeServiceLogsResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ListNodeServiceLogs xsi:type="soapenc:Array"
soapenc:arrayType="ns2:NodeServiceLogList[2]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
    <item>
     <name xsi:type="xsd:string">172.19.240.92</name>
     <ServiceLog xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[40]">
      <item>Cisco Syslog Agent</item>
      <item>Cisco Unified CM SNMP Service</item>
      <item>Cisco CDP Agent</item>
      <item>Cisco CDP</item>
      <item>Cisco Log Partition Monitoring Tool</item>
      <item>Cisco RIS Data Collector</item>
      <item>Cisco AMC Service</item>
      <item>Cisco Serviceability Reporter</item>
      <item>Cisco Unified CM Admin Web Service</item>
      <item>Cisco Unified CM Realm Web Service</item>
      <item>Cisco Unified CM Service Web Service</item>
      <item>Cisco SOAP Web Service</item>
      <item>Cisco RTMT Web Service</item>
      <item>Cisco CAR Web Service</item>
      <item>Cisco Unified CM PD Web Service</item>
      <item>Cisco Unified CM DBL Web Library</item>
      <item>Cisco Unified CM NCS Web Library</item>
      <item>Unified CM</item>
      <item>Cisco Unified IP Phone Services</item>
      <item>Cisco AXL Web Service</item>
      <item>Cisco WebDialer Web Service</item>
      <item>Cisco WebDialerRedirector Web Service</item>
      <item>Cisco Ccmuser Web Service</item>
      <item>Cisco Extended Functions</item>
      <item>Cisco CDR Repository Manager</item>
      <item>Cisco CDR Agent</item>
      <item>Cisco CAPF</item>
      <item>Cisco CTLProvider</item>
      <item>Unified CM</item>
      <item>Cisco DirSync</item>
      <item>Cisco CTIManager</item>
      <item>Cisco TFTP</item>
      <item>Cisco Ip Voice Media Streaming App</item>
      <item>CMI</item>
      <item>Cisco Database Layer Monitor</item>
      <item>Cisco Car Scheduler</item>
      <item>Cisco Ipma Services</item>
      <item>Cisco Extension Mobility</item>
      <item>Database Layer (DBL) Logs</item>
```

```
      <item>Prog Logs</item>
      <item>SQL DBMS Logs</item>
     </ServiceLog>
     <Systemlog xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[5]">
      <item>Event Viewer-Application Log</item>
      <item>Install Logs</item>
      <item>Event Viewer-System Log</item>
      <item>Security Logs</item>
      <item>Cisco Tomcat</item>
     </Systemlog>
   </ListNodeServiceLogs>
  </ns1:ListNodeServiceLogsResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Fault

If the database is not up and running or no node exists in the database, it will return a remote exception, "Query selectAllProcessNodes failed."

If no service list or syslog list is returned from the preferences, the system returns a remote exception: "No service found" or "No System Log found."

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:ListNodeServiceLogsResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <ListNodeServiceLogs xsi:type="soapenc:Array"
soapenc:arrayType="ns2:NodeServiceLogList[2]"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
    <item>
     <name xsi:type="xsd:string">172.19.240.92</name>
     <ServiceLog xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[40]">
      <item>Cisco Syslog Agent</item>
      <item>Cisco Unified CM SNMP Service</item>
      <item>Cisco CDP Agent</item>
      <item>Cisco CDP</item>
      <item>Cisco Log Partition Monitoring Tool</item>
      <item>Cisco RIS Data Collector</item>
      <item>Cisco AMC Service</item>
      <item>Cisco Serviceability Reporter</item>
      <item>Cisco Unified CM Admin Web Service</item>
      <item>Cisco Unified CM Realm Web Service</item>
      <item>Cisco Unified CM Service Web Service</item>
      <item>Cisco SOAP Web Service</item>
      <item>Cisco RTMT Web Service</item>
      <item>Cisco CAR Web Service</item>
      <item>Cisco Unified CM PD Web Service</item>
      <item>Cisco Unified CM DBL Web Library</item>
      <item>Cisco Unified CM NCS Web Library</item>
      <item>Unified CM Cisco Unified IP Phone Services</item>
      <item>Cisco AXL Web Service</item>
      <item>Cisco WebDialer Web Service</item>
      <item>Cisco WebDialerRedirector Web Service</item>
```

```
      <item>Cisco Ccmuser Web Service</item>
      <item>Cisco Extended Functions</item>
      <item>Cisco CDR Repository Manager</item>
      <item>Cisco CDR Agent</item>
      <item>Cisco CAPF</item>
      <item>Cisco CTLProvider</item>
      <item>Unified CM</item>
      <item>Cisco DirSync</item>
      <item>Cisco CTIManager</item>
      <item>Cisco TFTP</item>
      <item>Cisco Ip Voice Media Streaming App</item>
      <item>CMI</item>
      <item>Cisco Database Layer Monitor</item>
      <item>Cisco Car Scheduler</item>
      <item>Cisco Ipma Services</item>
      <item>Cisco Extension Mobility</item>
      <item>Database Layer (DBL) Logs</item>
      <item>Prog Logs</item>
      <item>SQL DBMS Logs</item>
     </ServiceLog>
     <Systemlog xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[5]">
      <item>Event Viewer-Application Log</item>
      <item>Install Logs</item>
      <item>Event Viewer-System Log</item>
      <item>Security Logs</item>
      <item>Cisco Tomcat</item>
     </Systemlog>
   </ListNodeServiceLogs>
  </ns1:ListNodeServiceLogsResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

LogCollectionPort Service: selectLogFiles Operation

Description

The selectLogFiles operation retrieves log files based on a selection criteria. This API takes FileSelectionCriteria object as an input parameter and returns the file name and location for that object.

The LogCollectionService URL is
http://hostname/logcollectionservice/services/LogCollectionPort

Parameters

The selectLogFiles operation includes the following elements:

- ServiceLogs—Array of strings. The available service options depends on the services that are activated on the Cisco Unified CM. The actual available options are as those returned by the listNodeServiceLogs operation at run time. For example:
  - Cisco Syslog Agent
  - Cisco Unified CM SNMP Service
  - Cisco CDP Agent
- SystemLogs—Array of strings.

Note SystemLogs element is not available in Cisco Unified CM release 7.1.3, and therefore should be empty.

- JobType—The collection type. The available options are:
  - DownloadtoClient
  - PushtoSFTPServer

  If you select PushtoSFTPServer, then the following elements are also required:
  - IPAddress
  - UserName
  - Password
  - Port
  - Remote Download Folder
- SearchStr—A non-null string.
- Frequency—The frequency of log collection. The available options are:
  - OnDemand
  - Daily
  - Weekly
  - Monthly

Note Only OnDemand option is currently supported for Frequency element. The other options (Daily, Weekly, and Monthly) are applicable for schedule collection that is currently not supported.

- ToDate—The end date for file collection. Format is **mm/yy/dd hh:mm AM/PM.** The ToDate element is required if you use absolute time range.
  File collection time range can be absoulte or relative. If you prefer relative time range, then the following elements are required:
  - RelText
  - RelTime

  If you prefer absolute time range, then the following elements are required:
  - ToDate
  - FromDate
- FromDate—The start date for file collection. Format is **mm/yy/dd hh:mm AM/PM.** The FromDate element is required if you use absolute time range.
- RelText—The file collection time range. The available options are:
  - Week
  - Day
  - Month
  - Hours
  - Minutes

- RelTime—The file collection time value. Gives all files from the specified time up to present. The available range is 1 to 100.
  For example, if the RelText is “Day” and RelTime is 1, then we get all files modified in the previous one day.
- TimeZone—The time zone value. The format is **Client: (GMT ±*n*) *Name of the time zone*** where, n is the offset time of the specified time zone and GMT. For example:
  - Client: (GMT-0:0) Greenwich Mean Time
  - Client: (GMT-8:0) Pacific Standard Time
- Port—The port number of the node.
- IPAddress—The IP address of the node.
- UserName—The service administrator username for the node.
- Password—The service administrator password for the node.
- ZipInfo—Indicates whether to compress the files during collection. This element is applicable only for PushtoSFTPServer option. The avilable options are:
  - True—The files are compressed.
  - False—The files are not compressed.
- RemoteFolder—The remote folder where the files are to be uploaded. This option is used only if you choose to upload trace files to SFTP or FTP server.

Request Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:SelectLogFiles soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
   <FileSelectionCriteria href="#id0"/>
  </ns1:SelectLogFiles>
  <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:SchemaFileSelectionCriteria"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/LogCollection/">
   <ServiceLogs xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[45]">
    <item>Cisco Syslog Agent</item>
    <item>Event Viewer-Application Log</item>
    <item>Install Logs</item>
    <item>Event Viewer-System Log</item>
    <item>Security Logs</item>
   </ServiceLogs>

   <SystemLogs xsi:type="xsd:string" xsi:nil="true"/>

   <JobType href="#id2"/>
   <SearchStr xsi:type="xsd:string"/>
   <Frequency href="#id1"/>
   <ToDate xsi:type="xsd:string" xsi:nil="true"/>
   <FromDate xsi:type="xsd:string" xsi:nil="true"/>
   <TimeZone xsi:type="xsd:string">Client:(GMT-8:0)Pacific Standard Time</TimeZone>
   <RelText href="#id3"/>
   <RelTime xsi:type="xsd:byte">5</RelTime>
   <Port xsi:type="xsd:byte">0</Port>
```

```
   <IPAddress xsi:type="xsd:string">MCS-SD4</IPAddress>
   <UserName xsi:type="xsd:string" xsi:nil="true"/>
   <Password xsi:type="xsd:string" xsi:nil="true"/>
   <ZipInfo xsi:type="xsd:boolean">false</ZipInfo>
  </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns4:Frequency"
xmlns:ns4="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">OnDemand</multiRef>
    <multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns3:JobType"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">DownloadtoClient</multiRef>
    <multiRef id="id3" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns4:RelText"
xmlns:ns4="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Hours</multiRef>
 </soapenv:Body>
</soapenv:Envelope>
```

Response Example

The response returns a FileSelectionResult object, which contains the list of matching file names and their location in the server.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Body>
<ns1:SelectLogFilesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
<FileSelectionResult xsi:type="ns2:SchemaFileSelectionResult"
xmlns:ns2="http://cisco.com/ccm/serviceability/soap/LogCollection/">
<Node xsi:type="ns2:Node">
<name xsi:type="xsd:string">MCS-SD4</name>
<ServiceList soapenc:arrayType="ns2:ServiceLogs[1]" xsi:type="soapenc:Array"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
<item xsi:type="ns2:ServiceLogs">
<name xsi:type="xsd:string" xsi:nil="true"/>
<SetOfFiles soapenc:arrayType="ns2:file[5]" xsi:type="soapenc:Array">
<item xsi:type="ns2:file">
<name xsi:type="xsd:string">syslogmib00000305.txt</name>
<absolutepath
xsi:type="xsd:string">/var/log/active/cm/trace/syslogmib/sdi/syslogmib00000305.txt</absolu
tepath>
<filesize xsi:type="xsd:string">2097082</filesize>
<modifiedDate xsi:type="xsd:string">Thu Jan 29 04:14:05 PST 2009</modifiedDate>
</item>
<item xsi:type="ns2:file">
<name xsi:type="xsd:string">syslogmib00000306.txt</name>
<absolutepath
xsi:type="xsd:string">/var/log/active/cm/trace/syslogmib/sdi/syslogmib00000306.txt</absolu
tepath>
<filesize xsi:type="xsd:string">2097083</filesize>
<modifiedDate xsi:type="xsd:string">Thu Jan 29 05:41:26 PST 2009</modifiedDate>
</item>
<item xsi:type="ns2:file">
<name xsi:type="xsd:string">syslogmib00000307.txt</name>
<absolutepath
xsi:type="xsd:string">/var/log/active/cm/trace/syslogmib/sdi/syslogmib00000307.txt</absolu
tepath>
<filesize xsi:type="xsd:string">2096868</filesize>
```

```
<modifiedDate xsi:type="xsd:string">Thu Jan 29 07:08:56 PST 2009</modifiedDate>
</item>
<item xsi:type="ns2:file">
<name xsi:type="xsd:string">syslogmib00000308.txt</name>
<absolutepath
xsi:type="xsd:string">/var/log/active/cm/trace/syslogmib/sdi/syslogmib00000308.txt</absolu
tepath>
<filesize xsi:type="xsd:string">2096838</filesize>
<modifiedDate xsi:type="xsd:string">Thu Jan 29 08:36:17 PST 2009</modifiedDate>
</item>
<item xsi:type="ns2:file">
<name xsi:type="xsd:string">syslogmib00000309.txt</name>
<absolutepath
xsi:type="xsd:string">/var/log/active/cm/trace/syslogmib/sdi/syslogmib00000309.txt</absolu
tepath>
<filesize xsi:type="xsd:string">100657</filesize>
<modifiedDate xsi:type="xsd:string">Thu Jan 29 08:40:20 PST 2009</modifiedDate>
</item>
</SetOfFiles>
</item>
</ServiceList>
</Node>
</FileSelectionResult>
<ScheduleList soapenc:arrayType="ns3:Schedule[0]" xsi:type="soapenc:Array"
xmlns:ns3="http://cisco.com/ccm/serviceability/soap/LogCollection/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"/>
</ns1:SelectLogFilesResponse>
</soapenv:Body>
</soapenv:Envelope>
```

Fault

If the specified frequency is null, it will throw a remote exception, “LogCollection frequency is null.” If the array of ServiceLogs and System Logs is null, it throws a remote exception, “No Service/Syslog are provided for the collection.” If a matching file is not found, it throws a remote exception, “The File Vector from the server is null.”

CDRonDemand SOAP Service

The CDRonDemand SOAP service comprises a public SOAP/HTTPS interface that is exposed to third-party billing applications or customers to allow them to query the Unified CM CDR Repository Node to retrieve CDR/CMR files on demand through the use of two new API calls, get_file_list and get_file.

In previous releases, the CDR database stored CDR records, and third-party applications could query the database directly for the CDR records. In this release, CDRs no longer get stored in the CDR database, but as flat files.

The CDR On-Demand Service allows applications to obtain CDR files in a two-step process. First, the application requests CDR file lists based on a specific time interval; then, it can request specific CDR files from those lists that are returned via as (s)FTP session.

The billing application can acquire a list of CDR files that match a specified time interval (get_file_list), with the maximum time span being 1 hour. If the application needs to retrieve CDR files that span an interval over 1 hour, multiple get_file_list requests must be made to the servlet.

After the list of files is retrieved, the third-party application can then request a specific file (get_file). Upon receiving the request, the servlet initiates a (s)FTP session and sends the requested file to the application. Only one file per request is allowed, to avoid timeouts and other potential complications.

The CDR Repository node normally transfers CDR files to the billing servers once on a preconfigured schedule, then deletes them per the Unified CM configuration and other criteria. If for some reason the billing servers do not receive the CDR files, or want to have them sent again, they can do so using the SOAP/HTTPS CDR On-Demand APIs. After CDR files are deleted, you cannot retrieve them.

The CDR On-Demand Service provides the following features:

- API to get a list of files that match a specified time interval (get_file_list)
- API to request a specific file that matches a specified filename (get_file)
- Limit of 1300 file names get returned from the get_file_list API
- Specified time range cannot span over 1 hour
- Service not available during CDR repository file maintenance window
- CDR files are sent via standard FTP or (s)FTP, which is user configurable
- API to request specific file (get_file) can return only one file per request
- Servlet needs to be activated through Service Activation Page

Before an application can access the CDR files, you must follow these steps to ensure that the SOAP APIs are activated from the Service Activation window on the CDR Repository Node where the CDR Repository Manager is activated:

Step 1 Go to http://**<IP Address of Unified CM node>:8080/ccmservice**

Step 2 Choose **Tools > Service Activation**.

Step 3 Select the server where the CDR Repository Manager resides.

Step 4 Under the CDR Services section, start the following services:

- Cisco SOAP - CDRonDemandService
- CDR Repository Manager

The CDR On-Demand Service depends on the CDR Repository Manager, so both must be activated.

Step 5 Click **Update** and wait until the page refreshes.

Table 4-8 provides a summary of the SOAP CDRonDemand service operations.

Table 4-8 SOAP CDRonDemandService Operations

Operation	Description	Reference
get_file_list	Allows an application to query the CDR Repository Node for a list of all the files that match a specified time interval	CDRonDemand Service: get_file_list Operation, page 4-131
get_file	Allows customers to request a specific CDR file that matches the specified filename	CDRonDemand Service: get_file Operation, page 4-133

Tip The On-Demand Service will not function during the maintenance window, which occurs every hour by default (this setting is configurable). The maintenance window generally runs from 10 seconds to a few minutes. Applications should submit requests outside of the maintenance window.

WSDL Definition

The following section provides the WSDL definition of the SOAP CDRonDemand serviece APIs:

```
<?xml version="1.0" encoding="UTF-8" ?>
<wsdl:definitions targetNamespace="http://schemas.cisco.com/ast/soap/"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://schemas.cisco.com/ast/soap/"
xmlns:intf="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<!--
WSDL created by Apache Axis version: 1.2RC3
Built on Feb 28, 2005 (10:15:14 EST)
-->
<wsdl:types>
    <schema targetNamespace="http://schemas.cisco.com/ast/soap/"
xmlns="http://www.w3.org/2001/XMLSchema">
        <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
        <complexType name="ArrayOfFileName">
                 <sequence>
                <element name="FileName" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
            </sequence>
        </complexType>
    </schema>
</wsdl:types>

<wsdl:message name="get_fileResponse" />

<wsdl:message name="get_file_listResponse">
    <wsdl:part name="get_file_listReturn" type="impl:ArrayOfFileName" />
</wsdl:message>

<wsdl:message name="get_file_listRequest">
    <wsdl:part name="in0" type="xsd:string" />
    <wsdl:part name="in1" type="xsd:string" />
    <wsdl:part name="in2" type="xsd:boolean" />
</wsdl:message>

<wsdl:message name="get_fileRequest">
    <wsdl:part name="in0" type="xsd:string" />
    <wsdl:part name="in1" type="xsd:string" />
    <wsdl:part name="in2" type="xsd:string" />
    <wsdl:part name="in3" type="xsd:string" />
    <wsdl:part name="in4" type="xsd:string" />
    <wsdl:part name="in5" type="xsd:boolean" />
</wsdl:message>

<wsdl:portType name="CDRonDemand">
    <wsdl:operation name="get_file_list" parameterOrder="in0 in1 in2">
        <wsdl:input message="impl:get_file_listRequest" name="get_file_listRequest" />
        <wsdl:output message="impl:get_file_listResponse" name="get_file_listResponse" />
    </wsdl:operation>
    <wsdl:operation name="get_file" parameterOrder="in0 in1 in2 in3 in4 in5">
        <wsdl:input message="impl:get_fileRequest" name="get_fileRequest" />
        <wsdl:output message="impl:get_fileResponse" name="get_fileResponse" />
    </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="CDRonDemandSoapBinding" type="impl:CDRonDemand">
```

```
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="get_file_list">
        <wsdlsoap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#CDRonDemand#get_file_list" />
        <wsdl:input name="get_file_listRequest">
                <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://schemas.cisco.com/ast/soap/" use="encoded" />
        </wsdl:input>
        <wsdl:output name="get_file_listResponse">
                <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://schemas.cisco.com/ast/soap/" use="encoded" />
        </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="get_file">
        <wsdlsoap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#CDRonDemand#get_file" />
        <wsdl:input name="get_fileRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:CDRonDemand" use="encoded" />
        </wsdl:input>
        <wsdl:output name="get_fileResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://schemas.cisco.com/ast/soap/" use="encoded" />
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>

<wsdl:service name="CDRonDemandService">
    <wsdl:port binding="impl:CDRonDemandSoapBinding" name="CDRonDemand">
        <wsdlsoap:address
location="https://SERVERNAME/CDRonDemandService/services/CDRonDemand" />
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

Security Considerations for CDRonDemand Service

You can use standard FTP or SFTP to deliver the CDR files. Refer to RFC959 and RFC2228 for further details about these applications.

The CDR On-Demand Service will create either a standard FTP or SFTP session with the billing server each time that a CDR file is to be sent. Exceptions get thrown whenever an error occurs on the Servlet side. In addition, all errors will get written into log files.

On the billing application side, Cisco recommends that billing applications implement code to catch these exceptions and display the exception string for detailed error conditions.

CDRonDemand Service: get_file_list Operation

The get_file_list operation allows an application to query the CDR Repository Node for a list of all the files that match a specified time interval. The time interval of the request cannot exceed one hour. If you want a list of files that span more than the one hour time interval that is allowed, you must make multiple requests to the servlet to acquire multiple lists of filenames.

The get_file_list API returns an array of strings that contain the list of all the filenames that match the specified time interval. If no filenames exist that match the time range, the value that is returned from the API call is simply null. If any time errors are encountered, exceptions get thrown. In addition, logs will be kept detailing the errors. Find these log files in the /var/log/active/tomcat/logs/soap/log4j directory.

A limit of 1300 file names can be returned to the application as a result of a get_file_list API call. If the file list that is returned contains 1300 file names, but does not span the entire requested time interval, you should make additional requests with the start time of the subsequent requests as the time of the last file name that was returned in the previous request.

Parameters

The get_file_list API expects the following parameters:

- **Start Time**—Mandatory parameter that specifies the starting time for the search interval. The format is a string: YYYYMMDDHHMM. No default value exists.
- **End Time**—Mandatory parameter that specifies the ending time for the search interval. The format is a string: YYYYMMDDHHMM. No default value exists.

Note The time span between Start Time and End Time must constitute a valid interval, but be not longer than 1 hour.

- **Where to get the files from**— Mandatory parameter that tells the servlet whether to include those files that were successfully sent to the third-party billing servers. The format is boolean.
  - True = Include both files that were sent successfully and files that failed to be sent.
  - False = Send only the files that failed to be sent. Do not include files that were sent successfully.

Request

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:get_file_list soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <in0 xsi:type="xsd:string">200511161000</in0>
      <in1 xsi:type="xsd:string">200511161059</in1>
      <in2 href="#id0"/>
    </ns1:get_file_list>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="xsd:boolean"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">true</multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <soapenv:Body>
    <ns1:get_file_listResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <get_file_listReturn xsi:type="ns1:ArrayOfFileName">
        <FileName
xsi:type="xsd:string">cdr_StandAloneCluster_01_200807081000_50</FileName>
        <FileName
xsi:type="xsd:string">cdr_StandAloneCluster_01_200807081001_51</FileName>
        <FileName
xsi:type="xsd:string">cdr_StandAloneCluster_01_200807081002_52</FileName>
      </get_file_listReturn>
    </ns1:get_file_listResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

CDRonDemand Service: get_file Operation

The get_file operation allows customers to request a specific CDR file that matches the specified filename. The resulting CDR file then gets sent to the customer via standard FTP or secure FTP, depending on the third-party billing application preference. The only constraint provides that the servlet can only process one file per request.

The get_file API returns normally with no value to indicate that the file has been successfully sent to the third-party billing server. If the transfer fails for any reason, exceptions get thrown. In addition, logs detailing the errors get saved in the /var/log/active/tomcat/logs/soap/log4j directory.

Parameters

The get_file API expects the following parameters:

- **Host Name**—Mandatory parameter (string) that specifies the hostname of the third-party billing application server, information that the servlet needs to connect to the billing server to deliver the CDR files.
- **User Name**—Mandatory parameter (string) that specifies the username for the third-party billing application server, information that the servlet needs to connect to the billing server to deliver the CDR files.
- **Password**—Mandatory parameter (string) that specifies the password for the third-party billing application server, information that the servlet needs to connect to the billing server to deliver the CDR files.
- **Remote Directory**—Mandatory parameter (string) that specifies the remote directory on the third-party billing application server to that the CDR servlet is to send the CDR files.
- **File Name**—Mandatory parameter (string) that specifies the filename of the CDR file that the third-party billing application wants delivered from the CDR On-Demand Service.
- **Secure FTP**—Mandatory parameter (Boolean) that specifies whether to use standard FTP or secure SFTP to deliver the CDR files. This depends on the third-party billing application configuration and preferences.

Request

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:get_file soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="urn:CDRonDemand">
      <in0 xsi:type="xsd:string">citadel</in0>
      <in1 xsi:type="xsd:string">root</in1>
      <in2 xsi:type="xsd:string">citadelroot</in2>
      <in3 xsi:type="xsd:string">/tmp/</in3>
      <in4 xsi:type="xsd:string">cdr_cluster1_cm1_200511161018_9</in4>
      <in5 href="#id0"/>
    </ns1:get_file>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="xsd:boolean"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">true</multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:get_fileResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="urn:CDRonDemand"/>
  </soapenv:Body>
</soapenv:Envelope>
```

Fault

The CDR On-Demand Service throws exceptions when certain error conditions are met:

- The servlet gets used during the maintenance period.
- The values that are entered for starting and ending time do not reflect the correct length – 12 bytes in the format YYYYMMDDHHMM is the correct length.
- The starting and ending time spans an interval of more than 1 hour.
- The starting time is greater than or equal to the ending time (invalid interval).
- No files exist in the CDR Repository.
- The (s)FTP connection to the remote node did not get established.
- The (s)FTP application failed to send the files that the third-party billing application requested.

The exception string describes the error condition that the billing application can print with the toString() function.

DimeGetFileService SOAP Service

The DimeGetFileService service provides for obtaining log files through the standard Direct Internet Message Encapsulation (DIME) protocol.

DimeGetFileService SOAP Service: getOneFile Operation

The getOneFile operation takes as an input parameter the absolute file name of the file that you want to collect from the server.

The return value specifies the file name, but the file name will have an AXIS-specific name. After the file is downloaded, you must replace it with the actual file name that you got from the server.

This APIS is in a different service, and the service name is DimeGetFileService. The URL is: https://host:8443/logcollectionservice/services/DimeGetFileService.

Request Format

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetOneFile soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="DimeGetFileService">
   <FileName xsi:type="xsd:string">/var/log/active/syslog/messages</FileName>
  </ns1:GetOneFile>
 </soapenv:Body>
</soapenv:Envelope>
```

Response Format

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:GetOneFileResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="DimeGetFileService">
   <DataHandler href="cid:967B4FFE5D1E6F693815D4CA118E91D0"/>
  </ns1:GetOneFileResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Fault

None.

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/env<soapenv:Body>"
    <soapenv:Body>
        <ns1:GetOneFile soapenv:encodingStyle="http://schemas.xmlsoap.org/
            <FileName xsi:type=”xsd:string”>/var/log/active/cm/trace/ris/sdi/
```

```
        </ns1:GetOneFile>
    </soapenv:Body>
</soapenv:Envelope>
```

VersionService

getActiveVersion Request

```
<?xml    version="1.0"    encoding="UTF-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Header>
      <wsa:Action

xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">urn:getActiveVersion</wsa:Action>
      <wsa:MessageID

xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">uuid:c37f60b9-92eb-4c1a-8de7-873
b60adce19</wsa:MessageID>
      <wsa:ReplyTo
            xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">

<wsa:Address>http://se032c-94-154.cisco.com:9080/servlet/WSACallBackHandler</wsa:Address>
         <wsa:PortType
         xmlns:ns1="http://example.org">ns1:LocalPart</wsa:PortType>
      </wsa:ReplyTo>
      <wsa:To

xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">https://10.94.171.238/platform-servi
ces/services/VersionService.VersionServiceHttpSoap11Endpoint</wsa:To>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <getActiveVersion    xmlns="http://services.api.platform.vos.cisco.com"/>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

getActiveVersion Response

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
      <wsa:To>http://se032c-94-154.cisco.com:9080/servlet/WSACallBackHandler</wsa:To>
      <wsa:MessageID>urn:uuid:B8E556B192135F3CC31298997621402</wsa:MessageID>
      <wsa:Action>urn:getActiveVersionResponse</wsa:Action>
      <wsa:RelatesTo>uuid:455bc1c3-8267-4902-9e7e-8f47b6200ad1</wsa:RelatesTo>
   </soapenv:Header>
   <soapenv:Body>
      <ns:getActiveVersionResponse xmlns:ns="http://services.api.platform.vos.cisco.com">
         <ns:return xmlns:ax217="http://element.services.api.platform.vos.cisco.com/xsd"
xmlns:ax218="http://api.platform.vos.cisco.com/xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ax217:VersionResponse">
            <ax217:remoteMessages xsi:nil="true"/>
            <ax217:result>internal.request.complete</ax217:result>
            <ax217:version>8.6.0.98000-9005</ax217:version>
         </ns:return>
      </ns:getActiveVersionResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

Authentication

The following sections describe the authentication for the Serviceability XML API.

Note To improve performance, the SOAP server uses a timed finite cache of authentication and authorization information for up to 100 users. Cached information persists for up to 2 minutes.

Basic

Basic authentication uses base64 to encode and decode the credential information. Because base64 is a two-way function, which requires no key, this protocol represents an insecure clear-text authentication. Many platforms implement Basic authentication and most HTTP client agents support it. Basic authentication can be secure if encryption, such as SSL, is used.

Secure

When you install/upgrade Unified CM, the HTTPS self-signed certificate, httpscert.cer, automatically installs on the default website that hosts the Unified CM virtual directories.

Hypertext Transfer Protocol over Secure Sockets Layer (SSL), which secures communication between the browser client and the server, uses a certificate and a public key to encrypt the data that is transferred over the internet. HTTPS also ensures that the user login password transports securely via the web.

The following Unified CM applications support HTTPS, which ensures the identity of the server:

- Cisco Unified Communications Manager Administration
- Unified CM Serviceability
- Cisco Unified IP Phone User Option Pages
- Unified CM Bulk Administration
- Cisco Unified Communications Manager Auto-Register Phone Tool
- Cisco Unified Communications Manager CDR Analysis and Reporting
- Unified CM Trace Collection Tool
- Unified CM Real Time Monitoring Tool

For more information, refer to the *Cisco Unified Communications Manager Security Guide, Release 7.0*.

Authorization

Each LDAP user gets checked against an MLA configuration for permissions. If the LDAP user in basic authentication does not have permission to “Read and Execute” and does not have the “Standard CCM Admin Users” role, the access to SOAP APIs gets denied.

Developer Tools

Each of the following Web Services includes a separate JSP page that can be referenced during development as developer support:

- Real-time Service—https://<server>:8443/realtimeservice
- Performance Service—https://<server>:8443/perfmonservice
- ControlCenter Service—https://<server>:8443/controlcenterservice
- Log Collection Service—https://<server>:8443/logcollectionservice
- CDR On-Demand Service—https://<server>:8443/CDRonDemandService

Note You must enter the name of each Web Service exactly as shown above.

Each Web Service JSP page offers these options:

- View Deployed Web Services
- View <Web Service> WSDL
- SOAP Monitor

View Deployed Web Services

This option displays a window that is similar to Figure 4-1 for each Web Service. The *(wsdl)* links display the code for the item(s) that are listed. For more information about viewing the web services, see the *Unified CM Administration Guide*.

Figure 4-1 Deployed Web Services Window

View <Web Service> WSDL

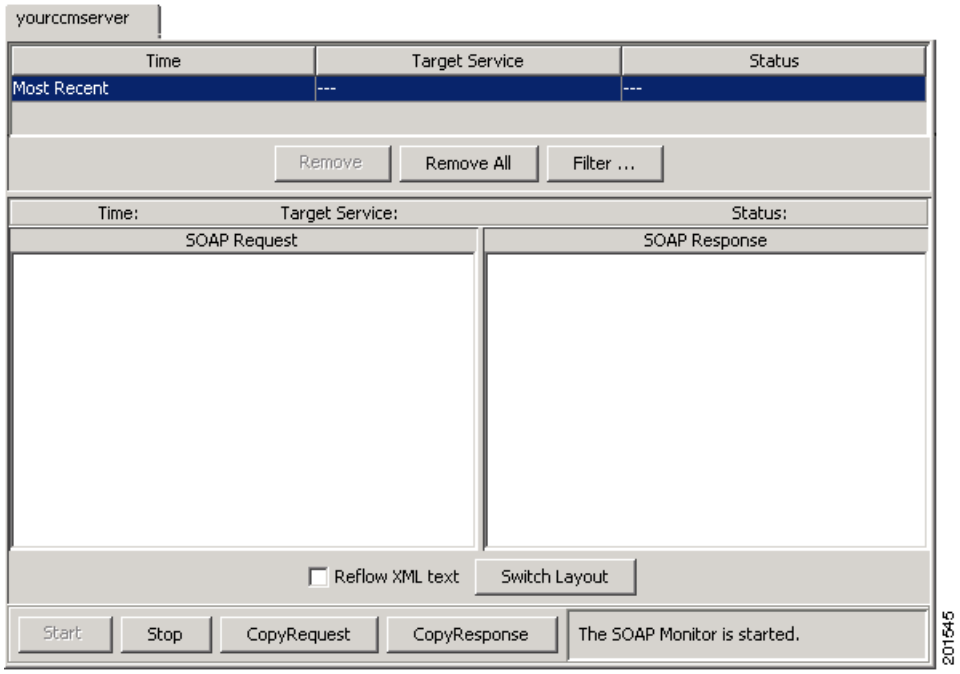
This option displays a window similar to Figure 4-2, which displays the WSDL code for the selected web service.

Figure 4-2 Web Service WSDL Window

SOAP Monitor

This option displays the SOAP Monitor Window, as shown in Figure 4-3. The SOAP Monitor window helps you look at the request and response messages during the development cycle.

Figure 4-3 SOAP Monitor Window



Password Expiration and Lockout

If a Credential Policy is defined for SOAP accounts to lock out with a count of login failures, the SOAP services returns “http 401 Unauthorized.”

If a Credential Policy is defined for SOAP accounts to expire, SOAP services returns “http 403 Forbidden.”

If a Credential Policy is defined for SOAP accounts to change the password, SOAP services returns “http 403 Forbidden.”

Application Customization for Cisco Unity Connection Servers

Be aware that some Serviceability SOAP operations are only available when the server runs the Unified CM software. Applications that support multiple server configurations (servers that run the Unified CM software, or the Cisco Unity Connection software, or both) must use the getProductInformation() interface to determine whether the operation they want to perform is available.

The following Serviceability SOAP operations are not available when only the Cisco Unity Connection software resides on the server:

Port or Service	Unavailable Operations
PerfmonPort	SelectCmDevice
CDROnDemand	*All operations*

The following Serviceability SOAP operations may provide different sets of target objects, depending on which software resides on the server:

Port or Service	Operations with Different Sets of Target Objects
PerfmonPort	perfmonAddCounter perfmonRemoveCounter perfmonCollectSessionData perfmonListInstance perfmonQueryCounterDescription perfmonListCounter perfmonCollectCounterData
ControlCenterServices	soapGetStaticServiceList soapGetServiceStatus soapDoServiceDeployment soapDoControlServices
LogCollectionPort	listNodeServiceLogs selectLogFiles DimeGetOneFile

SOAP Service Tracing

SOAP Service tracing is configurable through the Serviceability page in the Cisco Unified Serviceability application.

To configure debugging for the Cisco SOAP web service and the Cisco SOAPMessage service, follow these steps:

Step 1 From Cisco Unified Serviceability administration, choose **Trace > Configuration**.

Step 2 From the Server drop-down list, choose a server in your network.

Step 3 From the Service Group drop-down list, choose **Soap Services**.

Step 4 From the Service drop-down list, choose the service that you want to configure.

Step 5 If you want debugging to apply to all nodes, check the **Apply to All Nodes** check box.

Step 6 Make sure that the **Trace On** check box is checked.

Step 7 From the Debug Trace Level drop-down list, choose **Debug**.

Step 8 Click **Save**.

Step 9 Repeat this steps as needed to configure debugging for another service.

The traces can also be configured to debug level in Cisco Unified Serviceability administration. To do so, **Trace > Troubleshooting Trace Settings**, check the **Cisco SOAP Web Service** and **Cisco SOAP Message Service** check boxes, then click **Save**.

You can collect SOAP service logs for the Cisco SOAP web service and the Cisco SOAPMessage service by using the Unified CM Real Time Monitoring Tool (RTMT). For detailed instructions, refer to the RTMT on-line help.

Serviceability XML API Authentication Security

The HTTP transport that is used for the Serviceability XML API uses the following methods for authentication:

- Basic authentication—Uses base64 and SSL to encode and decode user name and password information. Because base64 is a two-way function and requires no key, this method is not secure. Basic authentication has the advantage of being widely implemented by many platforms and most HTTP client agents support it. Basic authentication is used with SSL to ensure a secure connection.
- Secure—When you install or upgrade Unified CM, HTTPS certificates are installed. Hypertext Transfer Protocol over SSL, which secures communication between the browser client and the server, uses a certificate and a public key to encrypt data that is transferred over the internet. HTTPS also ensures that the user log in password transports securely via the web.

Rate Control Mechanism

The Serviceability XML interface includes a request rate control mechanism that monitors the request rates for each server. If the request rate for a server is more than supported, a SOAP Fault is sent by the SOAP service on that server to the client and the additional requests are blocked. This rate control is enabled for Real-time Data Requests and Perfmon Data Requests. Rates are controlled through the enterprise parameters that are described in Table 4-9.

Table 4-9 Enterprise Parameters for Rate Control

Enterprise Parameter	Description
Allowed Performance Queries Per Minute	Specifies the maximum number of AXL performance counter queries that are allowed per minute for each server. Clients such as Voice Health Monitoring and Gateway Statistic Utility (GSU) receive a slow response if applications send more queries than the limit that is imposed by this parameter. Default value: 50 Minimum value: 1 Maximum value: 80

Table 4-9 Enterprise Parameters for Rate Control (continued)

Enterprise Parameter	Description
Allowed Device Queries Per Minute	Specifies the maximum number of AXL device queries that are allowed per minute for each server. Clients such as Voice Health Monitoring and Gateway Statistic Utility (GSU) receive a slow response if applications send more queries than the limit that is imposed by this parameter. Default value: 15 Minimum value: 1 Maximum value: 18
Maximum Performance Counters Per Session	Specifies the maximum number of performance counters that are allowed in a session-based request. Default value: 100 Minimum value: 0 Maximum value: 1000
Allowed CDRonDemand get_file Queries Per Minute	Specifies the maximum number of CDRonDemand get_file queries that are allowed per minute for each server. Default value: 10 Minimum value: 1 Maximum value: 20
Allowed CDRonDemand get_file_list Queries Per Minute	Specifies the maximum number of CDRonDemand get_file_list queries that are allowed per minute for the each server. Default value: 20 Minimum value: 1 Maximum value: 40

SOAP Fault Error Codes

Unified CM sends the AxisFaults to soap clients.

For general information about SOAP AxisFaults, refer to information provided by Apache for Apache Axis 1.4.

Fault Strings

The SOAP FaultString element contains the error string from Unified CM. Table 4-10 describes these error messages.

Table 4-10 Fault Strings in the SOAP FaultString Element

Error Message	Description	Error Location
Exceeded allowed rate for Realtime information. Current allowed rate for realtime information is " + maxcountperminute " requests per minute." +SOAPaction	Client has exceeded the configured limit for real-time SOAP requests that is configured for the rate control mechanism. For related information, see the "Rate Control Mechanism" section on page 4-143 section on page 2-97.	Client
"Exceeded allowed rate for CDRonDemand get_file_list. Current allowed rate is " +maxCdrGetFileListCountPerMinute +" requests per minute." + SOAPaction	Client has exceeded the configured limit for CDR on demand SOAP requests that is configured for the rate control mechanism. For related information, see the "Rate Control Mechanism" section on page 4-143 section on page 2-97.	Client
ERROR not able to resolve localhost	Serviceability XML soap server unable able to resolve localhost. This there is configuration error in Unified CM.	Server
ERROR the cmSelectionCriteria is null	Selection criteria specified in the request cannot be null. SOAP clients must specify the selection criteria in the request.	Client
RISBEAN COULD NOT BE INSTANCIATED	Connection between servlet and RIS data collector could not be established.	Server
ERROR: Invalid Class " + classStr	Class criteria specified in the request is not a valid device class.	Client
ERROR: Invalid Status " + statusStr	Status criteria specified in the request is not a valid registration status	Client
ERROR: Invalid node " + node2search	Node specified in the request is not reachable	Client
ERROR: Invalid node " + node2search + " Host "	Error message provides a description.	Client
ERROR: SelectBy is null"	SelectBy criteria has not been specified in the request.	Client
ERROR: Invalid DeviceDownloadStatus	Download status specified in the request is not valid.	Client
ERROR: Invalid ip address	IP address is not in the valid format.	Client
ERROR ctiSelectionCriteria is null	Selection criteria specified in the request cannot be null. SOAP clients must specify the selection criteria in the request.	Client

Table 4-10 Fault Strings in the SOAP FaultString Element (continued)

Error Message	Description	Error Location
ERROR: Invalid empty Cti Class	CtiMgrClass criteria specified in the request cannot be empty.	Client
ERROR: Invalid SelectAppBy	SelectAppBy criteria specified in the request has an invalid string.	Client
ERROR: Invalid Cti Class " + cticlasshere	Error message provides a description.	Client
ERROR: Invalid Cti Status " + ctistatushere	Error message provides a description.	Client
ERROR: Invalid node " + node2search + " Host " + myHost	Error message provides a description.	Client
Error Context is null	Error message provides a description.	Server
Failure trying to get the Call object"	Error message provides a description.	Server
Server.Unauthorized	Error message provides a description.	Client
DB access to NodeNames failed	Error message provides a description.	Client
-> soap getFileDirectoryList IO exception->	Error message provides a description.	Server
GetOneFile response message context null	Error message provides a description.	Server
FileName is NULL	Error message provides a description.	Client
Error found in Adding counters: Error=" + error code + " ErrorMsg=" + Error String	The perfmonAddCounter API fault includes one of these ReturnCode values: 1: “Not Found” client error 2: “Invalid Request” client error 3: “Internal Error” server error 9: ‘Invalid Node Name” client error 10: “Not Ready” server error 11: “Remote RisDC Down” server error 12: “Remote RisDC Not Ready” server error 13: “Collection Disabled”; server error 14: “No Collector Available” server error 50: “Handle Not Found In Cache” client error 100: “Perfmon Error” client error 101: “Add Counter Error” client error 102: “Invalid Request For All Instance Sessison” client error 103: “Invalid Counter Format” client error 127: “Unknown Error” client/server error	—

Table 4-10 Fault Strings in the SOAP FaultString Element (continued)

Error Message	Description	Error Location
Error found in Removing counters" + errorString	The perfmonRemoveCounter API fault includes one of these ReturnCode values: 1: “Not Found” client error 2: “Invalid Request” client error 3: “Internal Error” server error 9: ‘Invalid Node Name” client error 10: “Not Ready” server error 11: “Remote RisDC Down” server error 12: “Remote RisDC Not Ready” server error 13: “Collection Disabled”; server error 14: “No Collector Available” server error 50: “Handle Not Found In Cache” client error 100: “Perfmon Error” client error 101: “Add Counter Error” client error 102: “Invalid Request For All Instance Sessison” client error 103: “Invalid Counter Format” client error 127: “Unknown Error” client/server error	—

Table 4-10 Fault Strings in the SOAP FaultString Element (continued)

Error Message	Description	Error Location
Query counter failed. Error=" + Error code + " ErrorMsg=" + Error String	The perfmonCollectSessionData, perfmonListInstance, perfmonListCounter, or perfmonCollectCounterData API includes one of these ReturnCode values: 1: “Not Found” client error 2: “Invalid Request’ client error 3: “Internal Error” server error 9: “Invalid Node Name” client error 10: “Not Ready” server error 11: “Remote RisDC Down” server error 12: “Remote RisDC Not Ready” server error 13: “Collection Disabled”; server error 14: “No Collector Available” server error 50: “Handle Not Found In Cache” client error 100: “Perfmon Error” client error	—
ERROR: The ctiSelectionCriteria is null	Selection criteria specified in the request cannot be null. Soap Clients needs to specify this in the request.	Client
ERROR: Invalid SelectAppBy	SelectAppBy criteria specified in the request has an invalid string.	Client
ERROR: Invalid ip address	IP Address is not in the valid format.	Client
ERROR: Invalid DeviceDownloadStatus	DownloadStatus specified in the request is not valid.	Client

Sample SOAP Fault or AXIS Fault

The following example shows the format of a SOAP or AXIS fault.

```
<?xml version="1.0" encoding="UTF-8" ?> - <soapenv:Envelope
xmlns:soapenv="{*}http://schemas.xmlsoap.org/soap/envelope/* "
xmlns:xsd="http://www.w3.org/2001/XMLSchema "
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance ">- <soapenv:Body>- <soapenv:Fault>
  <faultcode>soapenv:Server.generalException</faultcode>
  <faultstring>{*}Error found in Adding counters: Error=101 ErrorMsg=\\suri-lab3\Memory%
Mem UUU;*</faultstring> - <detail>
  <ns1:stackTrace xmlns:ns1="{*}http://xml.apache.org/axis/* ">{*}Error found in Adding
counters: Error=101 ErrorMsg=\\suri-lab3\Memory% Mem UUU; at
com.cisco.ccm.serviceability.soap.perfport.PerfmonBindingImpl.perfmonAddCounter(Unknown
Source) at
com.cisco.ccm.serviceability.soap.perfport.PerfmonBindingSkeleton.perfmonAddCounter(Unknow
n Source) at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39) at
```

```
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at
java.lang.reflect.Method.invoke(Method.java:324) at
org.apache.axis.providers.java.RPCProvider.invokeMethod(RPCProvider.java:384) at
org.apache.axis.providers.java.RPCProvider.processMessage(RPCProvider.java:281) at
org.apache.axis.providers.java.JavaProvider.invoke(JavaProvider.java:319) at
org.apache.axis.strategies.InvocationStrategy.visit(InvocationStrategy.java:32) at
org.apache.axis.SimpleChain.doVisiting(SimpleChain.java:118) at
org.apache.axis.SimpleChain.invoke(SimpleChain.java:83) at
org.apache.axis.handlers.soap.SOAPService.invoke(SOAPService.java:454) at
org.apache.axis.server.AxisServer.invoke(AxisServer.java:281) at
org.apache.axis.transport.http.AxisServlet.doPost(AxisServlet.java:697) at
javax.servlet.http.HttpServlet.service(HttpServlet.java:709) at
org.apache.axis.transport.http.AxisServletBase.service(AxisServletBase.java:327) at
javax.servlet.http.HttpServlet.service(HttpServlet.java:802) at
sun.reflect.GeneratedMethodAccessor190.invoke(Unknown Source) at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at
java.lang.reflect.Method.invoke(Method.java:324) at
org.apache.catalina.security.SecurityUtil$1.run(SecurityUtil.java:239) at
java.security.AccessController.doPrivileged(Native Method) at
javax.security.auth.Subject.doAsPrivileged(Subject.java:500) at
org.apache.catalina.security.SecurityUtil.execute(SecurityUtil.java:268) at
org.apache.catalina.security.SecurityUtil.doAsPrivilege(SecurityUtil.java:157) at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.ja
va:231) at
org.apache.catalina.core.ApplicationFilterChain.access$000(ApplicationFilterChain.java:50)
at org.apache.catalina.core.ApplicationFilterChain$1.run(ApplicationFilterChain.java:140)
at java.security.AccessController.doPrivileged(Native Method) at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:136)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:214) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:104) at
org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:520) at
org.apache.catalina.core.StandardContextValve.invokeInternal(StandardContextValve.java:198
) at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:152)
at org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:104)
at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:540)
at org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:102)
at org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:520) at
org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:137) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:104) at
org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:118) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:102) at
org.apache.catalina.valves.AccessLogValve.invoke(AccessLogValve.java:535) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:102) at
org.apache.catalina.authenticator.SingleSignOn.invoke(SingleSignOn.java:417) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:102) at
org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:520) at
org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109) at
org.apache.catalina.core.StandardValveContext.invokeNext(StandardValveContext.java:104) at
org.apache.catalina.core.StandardPipeline.invoke(StandardPipeline.java:520) at
org.apache.catalina.core.ContainerBase.invoke(ContainerBase.java:929) at
org.apache.coyote.tomcat5.CoyoteAdapter.service(CoyoteAdapter.java:160) at
org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:799) at
org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.processConnection(Http11Pr
otocol.java:705) at
org.apache.tomcat.util.net.TcpWorkerThread.runIt(PoolTcpEndpoint.java:577) at
org.apache.tomcat.util.threads.ThreadPool$ControlRunnable.run(ThreadPool.java:683) at
java.lang.Thread.run(Thread.java:534)*</ns1:stackTrace>
  <ns2:hostname xmlns:ns2="{*}http://xml.apache.org/axis/*
">suri-lab3.cisco.com</ns2:hostname>
  </detail>
  </soapenv:Fault>
  </soapenv:Body>
```

Serviceability XML Application Design Guidelines and Best Practices

The following sections provide guidelines and best practices for designing Serviceability XML applications.

Maintain HTTPs sessions and Connection Timeouts

All SOAP clients need to maintain an HTTPS session. The SOAP server normally sends a set-cookie in response to an authentication request. The client must maintain and HTTPS session by sending cookies in subsequent SOAP requests as described in RFC 2109.

Clients should handle the connection timeouts and read timeouts. This approach helps release the connections and helps the web server to reuse the connection thread for another request.

Send Perfmon Close Session

It is the responsibility of the application to send the PerfmonCloseSession SOAP API.

The client program can use the following operations from the Serviceability XML APIs:

- Perfmon Open Session
- Perfmon Add Counter
- Perfmon Remove Counter
- Perfmon Collect Session Data
- Perfmon Close Session

A session handle is needed by the client to perform the session-based perfmonListCounter operation. The session handle is a universally unique identifier and is used only once. Ensure that there are no duplicate handles. The server stores the open handles in its cache. Session handles are removed by a Unified CM server when any of the following conditions occur:

- Session has been idle time out is reached
- Service/Server gets restarted
- Server resource is tight

An application should issue and PerfmonCloseSession API when a session completes. This API releases the memory on server side.

Device Query Support for Large Clusters

The following guidelines apply for device queries in large Unified CM clusters:

- The selectCmDevice operation includes StateInfo as part of the response. The StateInfo string must be in a CDATA element and indicates the state of the Device server. Clients must provide string from the first request to the next request for the same selection criteria. Clients receive the response with the NoChange element set to "true" as part of CmNode, which indicates that the server information state has not changed from the previous state. If NoChange in the response set to "false," the server information has changed. In this case, clients must obtain real-time information from the server and update the client information on the devices.

  The following example shows the schema of the StateInfo within the SelectCmDevice operation:

```
<!-- SOAP AST Header -->
<message name="AstHeader"><part name="AstHeader" type="tns:AstHeader"/></message>
<!-- R1. SelectCmDevice -->
<message name="SelectCmDeviceInput"><part name="StateInfo" type="xsd:string"/><part
name="CmSelectionCriteria" type="tns:CmSelectionCriteria"/>
</message>
<message name="SelectCmDeviceOutput"><part name="SelectCmDeviceResult"
type="tns:SelectCmDeviceResult"/><part name="StateInfo" type="xsd:string"/>
</message>

<complexType name="SelectCmDeviceResult">
<sequence><element name="TotalDevicesFound" type="xsd:unsignedInt"/><element
name="CmNodes" type="tns:CmNodes"/>
</sequence>
</complexType>
<complexType name="CmNodes">
<complexContent><restriction base="SOAP-ENC:Array"><attribute ref="soapenc:arrayType"
wsdl:arrayType="tns:CmNode[]"/>
</restriction>
</complexContent>
</complexType>
<complexType name="CmNode">
<sequence>
<element name="ReturnCode" type="tns:RisReturnCode"/><element name="Name"
type="xsd:string"/><element name="NoChange" type="xsd:boolean"/>
<element name="CmDevices" type="tns:CmDevices"/>
</sequence>
</complexType>
```

- The response provides a maximum of 200 devices, as shown in the following response schema:

```
<complexType name="CmDevices"><complexContent>
<restriction base="SOAP-ENC:Array">
<sequence><element name="CmDevice" type="tns:CmDevice" minOccurs="0"
maxOccurs="200"/></sequence>
</restriction>
</complexContent>
</complexType>
```

  This approach limits the response buffer to the first 200 devices that are returned. To obtain information about all configured devices, clients must embed the device information obtained from the AXL-DB method "SelectItems" into the serviceability method "CmSelectionCriteria."

  The following example shows this approach:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <soapenv:Body>
    <ns1:SelectCmDevice soapenv:encodingStyle="http://schemas.xmlsoap.org/
soap/encoding/" xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <StateInfo xsi:type="xsd:string"/>
      <CmSelectionCriteria href="#id0"/>
    </ns1:SelectCmDevice>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:CmSelectionCriteria"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="http://schemas.cisco.com/ast/soap/">
    <MaxReturnedDevices xsi:type="xsd:unsignedInt">200</MaxReturnedDevices>
    <Class xsi:type="xsd:string">Phone</Class>
    <Model xsi:type="xsd:unsignedInt">255</Model>
    <Status xsi:type="xsd:string">Registered</Status>
    <NodeName xsi:type="xsd:string" xsi:nil="true"/>
    <SelectBy xsi:type="xsd:string">Name</SelectBy>
    <SelectItems soapenc:arrayType="ns2:SelectItem[200]" xsi:type="soapenc:Array">
      <item href="#id1"/>
    </SelectItems>
  </multiRef>
  <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:SelectItem" xmlns:ns3="http://schemas.cisco.com/ast/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
 <Item xsi:type="xsd:string">SEP0123456789ab</Item>
 ...
    <Item xsi:type="xsd:string"> SEP0123456789aa</Item>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>
```

The most efficient way to obtain the list of devices is to use “executeSQLQuery” in the AXL-DB API.

You also can use the SQL equivalent to “Select name from device where tkclass = 1” to obtain all phones, then iterate through the list sending 200 at a time to the Serviceability XML API.

By default, device requests per minute can not exceed 15 per minute by default to the Unified CM server. Client requests should be spaced so that they do not exceed this limit. If client requests exceed this limit, the server responds with a SOAP fault. The SOAP client can then adjust the request rate. These rates are expected to take less than 20% of CPU resource so that they do not affect the performance of Unified CM.

Example of Stateinfo in a Response

The following example shows the StateInfo String from a response that contains a CDATA[] section. State ID for each Node is specified as an attribute StateId="123456".

```
<StateInfo xsi:type="xsd:string">

     &lt;StateInfo&gt;&lt;Node Name=&quot;sa-cm2-1&quot;
     SubsystemStartTime=&quot;1136458877&quot; StateId=&quot;2&quot;
     TotalItemsFound=&quot;0&quot; TotalItemsReturned=&quot;0&quot;/&gt;&lt;Node
     Name=&quot;sa-cm2-2&quot; SubsystemStartTime=&quot;1136403259&quot;
     StateId=&quot;33&quot; TotalItemsFound=&quot;15&quot;
     TotalItemsReturned=&quot;15&quot;/&gt;&lt;Node Name=&quot;sa-cm2-6&quot;
     SubsystemStartTime=&quot;1135982895&quot; StateId=&quot;387&quot;
     TotalItemsFound=&quot;1&quot; TotalItemsReturned=&quot;1&quot;/&gt;&lt;/StateInfo&gt;

</StateInfo>
```

Respond and React to SOAP Faults

It is the responsibility of the application to respond and react to SOAP faults.

SOAP faults are sent according to the SOAP standard. For additional information, see the "SOAP Fault Error Codes" section on page 4-144.

Limit Request and Response Size in the Application Design

When an application uses a SOAP interface, the application must ensure that the size of the SOAP request and response does not exceed the 1 MB limit. If this limit is exceeded, review the application design to determine another solution.

Usage notes for applications:

- Web server level security filters are configured to deny requests that exceed limits that are configured in your security applications.
- Tomcat Webserver 5.x and later provides a 2 MB limit on request size.
- Number of Perfmon counters per sessions is limited to 1,000.

Number of Nodes in the Cluster

In large cluster, configure your application to point SOAP clients to individual servers that have server specific Perfmon counters.

SOAP Monitor Usage

The SOAP Monitoring Tool should not be used in a production system because this tool can affect performance. User this tool only be used in development or unit testing and rely on SOAP logs for troubleshooting production systems.

CHAPTER 5

Serviceability XML Operations by Release

Table 5-1 lists alphabetically the new, changed, and deprecated serviceability XML operations by release. It also lists operations that are under consideration or review (UCR). Operation details can be found in Chapter 4, "Serviceability XML Programming."

Table legend:

- : Supported
- : Not supported
- : Modified

Operations By Release

Table 5-1 Serviceability XML Operations by Cisco Unified Communications Manager Releases

SOAP Service	Operation	5.0	6.0	6.1	7.0	7.1	8.0(1)	8.5(1)	8.6(1)
CDRonDemand (All CDR APIs)	get_file	✓	✓	✓	✓	✓	✓	✓	✓
	get_file_list	✓	✓	✓	✓	✓	✓	✓	✓
ControlCenterServicesPort (All Service Control APIs)	getProductInformationList	✗	✓	✓	✓	✓	✓	✓	✓
	soapDoControlServices	✓	✓	✓	✓	✓	✓	✓	✓
	soapDoServiceDeployment	✓	✓	✓	✓	✓	✓	✓	✓
	soapGetServiceStatus	✓	✓	✓	✓	✓	✓	✓	✓
	soapGetStaticServiceList	✓	✓	✓	✓	✓	✓	✓	✓
DimeGetFileService (Getting Single File)	GetOneFile	✓	✓	✓	✓	✓	✓	✓	✓
LogCollectionPort (All Log Collection APIs)	listNodeServiceLogs	✓	✓	✓	✓	✓	✓	✓	✓
	selectLogFiles	✓	✓	✓	✓	✓	✓	✓	✓

Table 5-1 Serviceability XML Operations by Cisco Unified Communications Manager Releases (continued)

SOAP Service	Operation	5.0	6.0	6.1	7.0	7.1	8.0(1)	8.5(1)	8.6(1)
PerfmonPort (Performance Information Port)	perfmonAddCounter	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonCloseSession	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonCollectCounterData	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonCollectSessionData	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonListCounter	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonListInstance	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonOpenSession	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonQueryCounterDescription	✓	✓	✓	✓	✓	✓	✓	✓
	perfmonRemoveCounter	✓	✓	✓	✓	✓	✓	✓	✓
RisPort (Real Time Informtion Port)	getServerInfo	✓	✓	✓	✓	✓	✓	✓	✓
	selectCmDevice (ipv4 devices)	✓	✓	✓	✓	✓	✓	✓	✓
	SelectCmDevice (includes ipv6 devices)	✗	✗	✗	✗	✓	✓	ⓘ	✓
	selectCtiItem (includes ipv6 devices)	✗	✗	✗	✗	✓	✓	✓	✓
	selectCtiItem (ipv4 devices)	✓	✓	✓	✓	✓	✓	✓	✓

PART 3

Extension Mobility Service API

CHAPTER 6

Cisco Extension Mobility Service API

This chapter describes the Cisco Extension Mobility (Extension Mobility) service. It contains the following sections:

Overview

The Cisco Extension Mobility service, a feature of Cisco Unified Communications Manager (Unified CM), allows a device, usually a Cisco Unified IP Phone, to temporarily embody a new device profile, including lines, speed dials, and services. It enables users to temporarily access their individual Cisco Unified IP Phone configuration, such as their line appearances, services, and speed dials, from other Cisco Unified IP Phones. The Cisco Extension Mobility service works by downloading a new configuration file to the phone. Unified CM dynamically generates this new configuration file based on information about the user who is logging in.

The system associates each Cisco Extension Mobility user with a device profile through configuration. When a user logs in to a phone, the phone temporarily embodies the device profile for that user. The two primary functions of the Cisco Extension Mobility feature comprise authenticating the user who is logging in and generating the right configuration file from the user information.

You can view the device profile as a template for a physical device. The device profile defines the attributes of a device, but it does not associate with a physical phone. A device profile includes information such as the phone template, user locale, services to which the device is subscribed, and so on. Because it does not associate with a physical phone, it does not have information such as MAC address, location, and region. When a phone downloads a device profile, the phone retains its physical attributes such as MAC address, device location information, device CSS, and so on.

The Unified CM support for the Cisco Extension Mobility service comprises the Cisco Extension Mobility Application (EMApp) and the Cisco Extension Mobility Service (EMService) modules. The application and service modules, along with other Unified CM infrastructure such as the Database Layer (DBL), User directory (either internal or an external LDAP directory), and TFTP server, provide the Cisco Extension Mobility feature.

You can use the XML-based EMService API with your applications, so they can take advantage of Cisco Extension Mobility service functionality. For details about how to use the Cisco Extension Mobility service API, see the "Using the Cisco Extension Mobility API" section on page 6-4.

To successfully develop an application that uses the Cisco Extension Mobility service, you need to understand how the service operates and how your application fits into the Cisco Extension Mobility service. This chapter includes the high-level concepts that are important in understanding the Cisco Extension Mobility service

New and Changed Information

The following sections provide information on the changes in the Extension Mobility APIs in Unified CM release 8.6(1) and the previous releases:

New Information for Cisco Unified Communications Manager 8.6(1)

There are no changes in the Extension Mobility API in Unified CM release 8.6(1).

New and Changed Information in Previous Releases of Unified CM

The following section provides the new and changed information in the older released of Unified CM release.

New Information for Cisco Unified Communications Manager 8.5(1)

There are no changes in the Extension Mobility API in Unified CM release 8.5(1).

New Information for Cisco Unified Communications Manager 8.0(1)

A new service parameter, called Maximum Concurrent Call Requests, is adde

The following sections describe API updates made in Unified CM 8.0(1):

New APIs

Table 6-1describes new APIs added in Unified CM 8.0(1).

Table 6-1 New APIs in Cisco Unified Communication Manager 8.0 (1)

APIs	Description
LogoutAll	This API enables the administrator to log out all users who are currently logged in.
DeviceProfileQuery	This API gets information of all the device profiles of a user.

From Unified CM release 8.0(1) onwards the Extension Mobility APIs support secure communication using HTTPS. As a result of this enhancement all communication between the various EM entities (device to EMApp, EMApp to EMService, 3rd party App to EM Service) is encrypted. But to provide backward compatibility, devices that do not support secure communications could continue to communicate with EM by using HTTP.

Note The Extension Mobility APIs do not support the Extension Mobility Cross Cluster feature.

Changed APIs

All APIs support communication using HTTPS.

How Cisco Extension Mobility Works

This section describes what happens when your application sends a message to the Extension Mobility service to use its functionality.

Your login application submits an XML message to the EMService servlet by using the Hypertext Transfer Protocol (HTTP) or Hypertext Transfer Protocol Secure (HTTPS). The EMService uses the LDAP directory to check the UserID and PIN in the message from the login application. If the UserID and PIN are valid, the EMService executes the request by communicating with the database layer (DBL) through JNI. For more details about how the Extension Mobility module works, see the Device Profiles section that follows.

If the DBL changes the device profile for a login or logout request, it tells the DBL Monitor, which passes this information on to the CallProcessing and CTI components. CallProcessing, in turn, tells the Cisco Unified IP Phone that it needs to restart itself to load the new device profile. For more information about device profiles, see the Device Profiles, page 6-4.

The CTI layer notifies JTAPI and TAPI applications that are monitoring the device or the user that the application control list has changed.

If the DBL completes a transaction successfully the information is passed to the EMService. The EMService then sends an XML response that the transaction was successful to your login application by using HTTP or HTTPS.

If the transaction is not successful, the EMService sends your login application an appropriate error message.

Device Profiles

Device profiles act as the basic unit of transaction for the Cisco Extension Mobility service. A device profile contains all the configuration information, such as line appearances, speed dials, and services, for a particular device. You can think of it as a virtual device. It has all the properties of a device except physical characteristics such as a Media Access Control (MAC) address and a directory URL.

When a user logs in, the User device profile replaces the current device configuration. When a user logs out, the Logout device profile replaces the User device profile.

Cisco Extension Mobility requires a Logout Device Profile for each configured device. Cisco Extension Mobility uses the Logout Device Profile, which can be either the current device settings or the User Device Profile, as the logged out configuration of the device.

Note Cisco Extension Mobility fully supports Cisco Unified IP Phone 7960 and Cisco Unified IP Phone 7940, but not Cisco Unified IP Phone model 7910 and preceding devices.

Using the Cisco Extension Mobility API

The Cisco Extension Mobility service provides a fairly rich API, which enables extension mobility on Cisco Unified IP Phones and allows application control over authentication, scheduling, and availability.

An application that uses Cisco Extension Mobility service represents an IP phone service that allows a user to enter a userID and PIN at the phone itself and log in to the phone. The architecture and implementation of Cisco Extension Mobility make many other applications possible; some examples follow:

- An application that automatically activates phones for employees when they reserve a particular desk for a particular time (the scheduling application)
- A lobby phone that does not have a line appearance until a user logs in

The Cisco Extension Mobility API gets exposed as an Extensible Markup Language (XML) interface via HTTP or HTTPS. The administrator of the system designates a website as the entry point to the API, and all requests and queries are made through those URLs. This website also provides the document type definitions (DTDs) that define the XML for requests, queries, and responses. This document includes the DTDs, along with examples.

The XML input gets submitted via an HTTP or HTTPS POST. A field named “xml” contains the XML string that defines the request or query. The response to this HTTP POST represents a pure XML response with either a success or failure indicator for a request or the response to a query.

Note The Cisco Extension Mobility API does not use the M-POST method as defined in the HTTP Extension Framework.

This section includes the following topics:

Set Up and Configuration

The Cisco Extension Mobility service application accompanies Unified CM. As such, all necessary Cisco Extension Mobility service API components are installed with the standard Unified CM installation.

To use the Cisco Extension Mobility service, create a device profile for the user who is logging in and for the target device. Use the following steps to configure Cisco Extension Mobility service:

- Activate the service.
- Create Extension Mobility IP phone service.
- Create a user device profile.
- Assign the user device profile to a user.
- Assign authentication proxy rights to a UserID.
- Assign UserID to the Standard EM Authentication Proxy Rights user group.
- Enable Cisco Extension Mobility and configure the default device profile on the target device. (You must enable Cisco Extension Mobility on a device-by-device basis.)
- Subscribe to Cisco Extension Mobility IP phone service on the target device and the device profile.
- Assign a logout device profile to a target device.
- Configure the system parameters (the system uses defaults if parameters are not manually configured).

Note Technically, no need exists to assign a profile to a user. The device profile can be specified at login.

For details on how to configure the User Device Profile, refer to the *Cisco Unified Communications Manager Administration Guide* or *Cisco Unified Communications Manager Features and Services Guide*.

Messages

You communicate between your login application and the Cisco Extension Mobility service by sending and receiving XML messages. The XML messages that you send must follow the rules that are set by the Message DTDs that are described in the Message Document Type Definitions, page 6-6.

The default URL for login and logout requests and system queries is

https://<server>:8443/emservice/EMServiceServlet

If the device does not support secure communication then URL for login and logout requests and system queries is:

http://<server>:8080/emservice/EMServiceServlet

The application sends authentication information, including an Application ID and an Application Certificate, at the start of message.

A password represents the only type of certificate that is currently supported. All messages must include a valid appID and appPassword, or they do not get processed. For examples of valid Cisco Extension Mobility messages, see the Message Examples, page 6-8.

Login Requests

Login requests provide the cornerstone of this service, and currently they offer the most flexible and complex message type. The information that is required to process a login request must include the device that is to be logged in to and the UserID of the user who is logging in to that device. If a device profile other than the default device profile that has been associated with the user is to be used, you can specify that profile name. If the system is to automatically log the user out after a particular time, you can also specify that. To log out, you only need to provide the device name in the message.

Logout All

The LogoutAll API enables the administrator to logout all extension mobility users who are currently logged in.

Note Use of Logout All API may lead to generation of large number of database change notifications which may impact system performance momentarily. The number of change notification will be directly proportional to the number of users logged in. Thus it is recommended not to use this API in peak business hours.

Device-User Queries

A Device-User query represents a query wherein the login application specifies a list of one or more devices, and the system returns the userID of the user who is currently logged on to each device.

User-Devices Queries

A User-Devices query represents a query in which the login application specifies a list of one or more users, and the system returns the list of devices to which a particular user is currently logged in.

DeviceProfileQuery

The DeviceProfileQuery API gets information of all the device profiles associated to an user. The API response provides information about all the associated device profiles for an intra cluster user, that is end user in local DB, and also indicate if the profile is the default profile for the user.

Message Document Type Definitions

A Message Document Type Definition (DTD) designates an XML list that specifies precisely which elements can appear in a request, query, or response document. It also specifies the contents and attributes of the elements.

You communicate between your login application and the Cisco Extension Mobility service by sending and receiving XML documents. These XML documents must follow the rules that the Message DTDs set. For examples of how Message DTDs are used, see the Message Examples, page 6-8.

Request DTD

The Request DTD defines the login and logout messages that your application can send to the Cisco Exchange Mobility service.

```
<!-- login requests DTD -->
<!ELEMENT request (appInfo, (login | logout | logoutAll))>
<!ELEMENT appInfo (appID, (appCertificate | appEncryptedCertificate))>
<!ELEMENT appID                   (#PCDATA)>
<!ELEMENT appCertificate          (#PCDATA)>
<!ELEMENT appEncryptedCertificate (#PCDATA)>
<!ELEMENT login (deviceName, userID, deviceProfile?, exclusiveDuration?, remoteIPAddr?,
isViaHeaderSet?)>
<!ELEMENT logout (deviceName, remoteIPAddr?, isViaHeaderSet?)>
<!ELEMENT logoutAll (remoteIPAddr?, isViaHeaderSet?)>
<!ELEMENT deviceName     (#PCDATA)>
<!ELEMENT userID         (#PCDATA)>
<!ELEMENT deviceProfile  (#PCDATA)>
<!ELEMENT remoteIPAddr (#PCDATA)>
<!ELEMENT isViaHeaderSet (#PCDATA)>
<!ELEMENT exclusiveDuration (time | indefinite)>
<!ELEMENT time           (#PCDATA)>
<!ELEMENT indefinite EMPTY>
```

Login or Logout Response DTD

Login or Logout Response DTD defines the messages that your application receives from the Cisco Extension Mobility service when it sends a login or logout request message.

```
<!-- login response DTD -->
<!ELEMENT response (success | failure)>
<!ELEMENT success EMPTY>
<!ELEMENT failure (error)>
<!ELEMENT error (#PCDATA)>
<!ATTLIST error code NMTOKEN #REQUIRED>
```

Note

- The Clear Call Log service parameter is set to true to clear the call logs. But the call log is cleared only during the Extension Mobility manual logout process. If a logout occurs due to an automatic logout or any occurrence other than a manual logout, the call logs do not get cleared.
- To clear call logs:
  - Hard reset the device by setting the isHardReset parameter of the doDeviceReset AXL API to true OR
  - Send an Init:CallHistory uniform resource identifier (URI) via the IP Phone XML service interface

Query DTD

The Query DTD defines the Device-User and User-Devices messages that your application sends the Cisco Extension Mobility service to find out which user is logged in to a device or to which devices users are logged in.

```
<!-- login query DTD -->
<!ELEMENT query (appInfo, (deviceUserQuery | userDevicesQuery | checkUser |
deviceProfileQuery))>
```

```
<!ELEMENT appInfo (appID, (appCertificate | appEncryptedCertificate))>

<!ELEMENT appID                   (#PCDATA)>
<!ELEMENT appCertificate          (#PCDATA)>
<!ELEMENT appEncryptedCertificate (#PCDATA)>

<!ELEMENT deviceUserQuery (deviceName+,remoteIPAddr?)>
<!ELEMENT userDevicesQuery (userID+,remoteIPAddr?)>
<!ELEMENT checkUser (userID+,remoteIPAddr?,isViaHeaderSet?)>
<!ELEMENT deviceProfileQuery (userID+,remoteIPAddr?,isViaHeaderSet?)>

<!ELEMENT deviceName     (#PCDATA)>
<!ELEMENT userID         (#PCDATA)>
<!ELEMENT remoteIPAddr   (#PCDATA)>
<!ELEMENT isViaHeaderSet (#PCDATA)>
```

Query Response DTD

The Query Response DTD defines the messages that your application receives from the Cisco Extension Mobility service when it sends the service a Device-User or User-Devices query.

```
<!-- login query results DTD -->
<!ELEMENT response (deviceUserResults | userDevicesResults | deviceProfileResults|
failure)>
<!ELEMENT deviceUserResults (device+)>
<!ELEMENT userDevicesResults (user+)>
<!ELEMENT deviceProfileResults (userID+)>
<!ELEMENT device (userID | lastlogin | none | doesNotExist)>
<!ATTLIST device
          name NMTOKEN #REQUIRED>
<!ELEMENT user (deviceName+ | none | doesNotExist)>
<!ATTLIST user
          id NMTOKEN #REQUIRED>
<!ELEMENT userID (#PCDATA)>
<!ELEMENT lastlogin (#PCDATA)>
<!ELEMENT deviceName (#PCDATA)>
<!ELEMENT none EMPTY>
<!ELEMENT doesNotExist EMPTY>
<!ELEMENT failure (errorMessage)>
<!ELEMENT errorMessage (#PCDATA)>
```

Message Examples

This section provides examples of various types of messages to aid in understanding how to use the message DTDs to communicate between your login application and the Cisco Extension Mobility service.

Login Operation

The Login operation logs in a single user using the specified device profile.

Sample Login Code

The following example logs in userID “john” to device SEP003094C25B15 using the User Device Profile UserDevProf:

```
<request>
```

```
        <appInfo>
                <appID>appid</appID>
                <appCertificate>apppasswd</appCertificate>
        </appInfo>
        <login>
                <deviceName>SEP003094C25B15</deviceName>
                <userID>john</userID>
                <deviceProfile>UserDevProf</deviceProfile>
                <exclusiveDuration>
                          <time>60</time>
                </exclusiveDuration>
        </login>
</request>
```

Success Response

```
<response>
  <success/>
</response>
```

Failure Response

```
<response>
  <failure>
    <error code="3">Could not authenticate 'appid'</error>
  </failure>
</response>
```

Logout Operation

The Logout operation logs out a single user from the specified device.

Sample Logout Code

The following example logs out a user who is logged into device SEP003094C25B15:

```
<request>
        <appInfo>
               <appID>appid</appID>
               <appCertificate>apppasswd</appCertificate>
        </appInfo>
        <logout>
               <deviceName>SEP003094C25B15</deviceName>
        </logout>
</request>
```

Success Response

```
<response>
  <success/>
</response>
```

Failure Response

```
<response>
  <failure>
    <error code="3">Could not authenticate 'appid'</error>
  </failure>
</response>
```

LogoutAll Operation

The LogoutAll operation logs out all the users from the device.

Sample LogoutAll Request

```
<request>
        <appInfo>
               <appID>appid</appID>
               <appCertificate>apppasswd</appCertificate>
        </appInfo>
        <logout>
               <deviceName>SEP003094C25B15</deviceName>
        </logout>
</request>
```

Sample LogoutALL Response

```
<request>
        <appInfo>
               <appID>appid</appID>
               <appCertificate>apppasswd</appCertificate>
        </appInfo>
        <logoutAll>
        </logoutAll>
</request>
```

UserQuery Operation

The UserQuery operation returns the user ID that is logged in to the specified device.

Sample UserQuery Request

The following example finds the user who is logged in to the device SEP003094C25B15:

```
<query>
        <appInfo>
                <appID>appid</appID>
                <appCertificate>apppasswd</appCertificate>
        </appInfo>
        <deviceUserQuery>
                <deviceName>SEP003094C25B15</deviceName>
        </deviceUserQuery>
</query>
```

Sample UserQuery Response

If you log in to the phone for the first time, the response is as follows:

```
<response>
<deviceUserResults>
<device name="SEP00016CEA6616">
<userID>one</userID>
<none/>
</device>
</deviceUserResults>
</response>
```

If you have previously logged in to the phone, the response is as follows:

```
<response>
<deviceUserResults>
<device name="SEP......">
```

```
<userID>one</userID>
<lastlogin>one</lastlogin>
</device>
</deviceUserResults>
</response>
```

DeviceQuery Operation

The DeviceQuery operation returns all device IDs (MAC addresses) for the specified user ID.

Sample DeviceQuery Request

The following example finds the devices that user ID “john” is logged in to:

```
<query>
        <appInfo>
                <appID>appid</appID>
                <appCertificate>apppasswd</appCertificate>
       </appInfo>
       <userDevicesQuery>
                <userID>john</userID>
       </userDevicesQuery>
</query>
```

Sample DeviceQuery Response

```
<response>
  <userDevicesResults>
  <user id="rknotts">
  <deviceName>SEP003094C25B15</deviceName>
  <deviceName>SEP003094C25B49</deviceName>
  </user>
  <user id="fwragge">
  <deviceName>SEP003094C249A6</deviceName>
  </user>
  </userDeviceResults>
</response>
```

Device Profile Query

The DeviceProfileQuery API gets information of all the device profiles associated to an user.

Sample Device Profile Query Request

```
<query>
        <appInfo>
                <appID>appid</appID>
                <appCertificate>apppasswd</appCertificate>
       </appInfo>
       <userDeviceProfileQuery>
                <userID>john</userID>
       </userDeviceProfileQuery>
</query>
```

Sample DeviceProfile Query Response

```
<results>
    <deviceProfileResults>
        <user id="john">
            <deviceProfileName>UDP1</deviceProfileName>
```

```
            <deviceProfileName>UDP2</deviceProfileName>
        </user>
    </deviceProfileResults>
</results>
```

Extension Mobility Service Response Codes

Table 6-2 describes the response codes and messages that can be returned by the Extension Mobility service. A response contains a code and a response string, formatted as an XML string.

Table 6-2 Cisco Extension Mobility Service Response Codes

Response Code	Message	Description
0	Unknown Error	Generic error, which does not belong to the known error types.
1	Error on Parsing	Invalid XML request. The XML passed does not conform to the DTD.
2	Cannot auth. App user	**Blank UserID or PIN** NULL_PARAM—Shows the user login page with error title.
3	Invalid App User	The appid supplied in the XML is not a valid user.
4	Policy Validation error	Does not conform to the policy set up for the user. For example, multiple log in not allowed.
5	Dev. logon disabled	Extension Mobility is not enabled on the device at the time of log out.
6	Database Error	Database is unable to process the Extension Mobility request.
7	Logout Request Error	Could not set auto-logout duration during log in. Could not remove the device from the auto-logout list after log out.
8	Query type undetermined	Unrecognized Query type. The query type provided in the XML is not supported.
9	Dir. User Info Error	Could not authenticate user. This error is a for various authentication related failures.
10	User lacks app proxy rights	If a userID also is used as an appID, the userID should have proxy rights.
11	Device does not exist	Trying to perform an operation on a device that does not exist.
12	Dev. Profile not found	Trying to use a User Device Profile that does not exist.
18	Another user logged in	Another user is logged in to the device where login is being performed.
19	No user logged in	Trying to perform log out on a device where no user is currently logged in.
20	Hoteling flag error	Could not retrieve the Extension Mobility “Enabled” status for the specified device (DB error).
21	Hoteling Status error	Could not verify the login status of the specified device (DB error).
22	Dev. logon disabled	Extension Mobility is not enabled on the device.

Table 6-2 Cisco Extension Mobility Service Response Codes (continued)

Response Code	Message	Description
23	User not found	Given user ID is invalid.
25	User logged in elsewhere	User is trying to log in to a device, but is already logged in to another device and multiple log in is not allowed.
26	Busy, please try again	The server currently is processing the maximum number of log in/log out requests. Additional requests will be accepted after pending ones are processes.
27	Change Password	Password must change on first use or password has expired. User must change password from the User page in Unified CM Administration.
28	Untrusted IP Error	Trying to log in or log out from an untrusted IP address.
29	ris down-contact admin	The RIS Data Collector service is down. An administrator must turn it on.
30	Proxy not allowed	Log in or Log out using a proxy server is not allowed.

CHAPTER 7

Cisco Extension Mobility Operations By Release

Table 7-1 lists alphabetically the new, changed, and deprecated Cisco Extension Mobility operations by release. It also lists operations that are under consideration or review (UCR). Operation details can be found in Chapter 6, "Cisco Extension Mobility Service API."

Table legend:

- —Supported
- —Not supported

Operations By Release

Table 7-1 Extension Mobility Operations by Cisco Unified Communications Manager Release

Operation	5.0(d)	5.1	5.1(2)	6.0	6.1	7.0	7.1(2)	7.1(3)	8.0(1)	8.5(1)	8.6(1)
DeviceProfileQuery	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
DeviceQuery	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Login	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Logout	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LogoutAll	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
UserQuery	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

PART 4

Web Dialer API

CHAPTER 8

Cisco Web Dialer API Programming

This chapter describes the Simple Object Access Protocol (SOAP) and HTML over secure HTTP (HTTPS) interfaces that are used to develop customized click-to-dial applications for Cisco Web Dialer (Web Dialer) for Cisco Unified Communications Manager (Unified CM) and contains the following sections:

Overview

Web Dialer is a service that can be activated on a Unified CM subscriber to enable custom developed click-to-dial applications to issue MakeCall requests on behalf of a user. These applications can be server based, such as a click-to-dial enabled corporate directory, or desktop-based, such as an Outlook plug-in that lets users click to dial contacts.

The two main components of Web Dialer are the Web Dialer servlet and the Redirector servlet.

Table 8-1 explains some terms that are used in this chapter.

Table 8-1 Web Dialer Terms

Cisco Web Dialer Service	A server-based component of Cisco Unified Communications Manager that allows users to make calls from web and desktop applications.
Cisco Web Dialer Application	A customer or partner created application that calls SOAP or HTTP methods from the Web Dialer interface library.

Table 8-1 Web Dialer Terms (continued)

Web Dialer Servlet	A Java servlet that responds to SOAP or HTTP requests.
Redirector Servlet	A Java servlet that finds the home Unified Communications Manager cluster of a user and responds with one or more IP addresses of the Web Dialer enabled subscribers within the home cluster.

New and Changed Information

The following sections provide information on the changes in the Web Dialer APIs in Unified CM release 8.6(1) and the previous releases:

- New Information for Cisco Unified Communications Manager 8.6(1), page 8-2
- New and Changed Information in Previous Releases of Unified CM, page 8-2

For information about new, changed, or deprecated Web Dialer API methods from the interface library, see Chapter 9, "Cisco Web Dialer Operations By Release."

New Information for Cisco Unified Communications Manager 8.6(1)

There are no changes in the Web Dialer APIs in Unified CM release 8.6(1).

New and Changed Information in Previous Releases of Unified CM

The following sections provide the new and changed information in the older releases of Unified CM:

- New Information for Cisco Unified Communications Manager 8.5(1), page 8-2
- New Information for Cisco Unified Communications Manager 8.0(1), page 8-3
- New Information for Cisco Unified Communications Manager 7.1(2), page 8-3
- New Information for Cisco Unified Communications Manager 7.0, page 8-3
- New Information for Cisco Unified Communications Manager 6.0, page 8-3
- New Information for Cisco Unified Communications Manager 5.1, page 8-3

For information about new, changed, or deprecated Web Dialer API methods from the interface library, see Chapter 9, "Cisco Web Dialer Operations By Release."

New Information for Cisco Unified Communications Manager 8.5(1)

The following change was done in the Web Dialer APIs in Unified CM release 8.5(1):

- The Maximum Concurrent Call Requests was raised from six to eight. For more information, see Maximum Concurrent Call Requests, page 8-7

New Information for Cisco Unified Communications Manager 8.0(1)

A new service parameter, called Maximum Concurrent Call Requests, is added for modifying the throttle value of Web Dialer requests. This value was previously hard-coded. The throttle limits the number of CTI requests from Web Dialer. The minimum and maximum values for this throttle are one and six, and the recommended values for 7825 and 7845 servers are three and six respectively.

For more information, see Maximum Concurrent Call Requests, page 8-7.

New Information for Cisco Unified Communications Manager 7.1(2)

There are no changes in Web Dialer for Unified CM 7.1(2).

New Information for Cisco Unified Communications Manager 7.0

The following SOAP API methods have been added for Web Dialer in Unified CM 7.0:

- getProfileDetailSoap
- getPrimaryLine

New Information for Cisco Unified Communications Manager 6.0

Unified CM 6.0 includes the following change to Web Dialer:

- The getProfilSoap method returns only devices that are supported by Web Dialer. These devices are derived from those that are supported by Cisco JTAPI. Devices that are not supported by Cisco JTAPI are no longer returned. For additional information, refer to *Cisco Unified Communications Manager JTAPI Developers Guide* for release 6.0(1), which is available at this URL:

  http://www.cisco.com/en/US/docs/voice_ip_comm/cucm/jtapi_dev/6_0_1/jtapi-dev.html

- Application Dial Rules support has been added for the SOAP API.

New Information for Cisco Unified Communications Manager 5.1

Unified CM 5.1 includes the following change to Web Dialer:

- Web Dialer and Redirector now require HTTPS.

  Developers should format Redirector and Web Dialer requests to use HTTPS. Unified CM requires the secured protocol to prevent unauthorized applications from reading user data.

Refer to *Cisco Unified CallManager Developers Guide for Release 5.0* for important changes to Web Dialer API programming in the 5.0 release.

Cisco Web Dialer Components

The following sections provide information about Web Dialer Components:

Cisco Web Dialer Servlet

The Web Dialer servlet, a Java servlet, allows Cisco Unified Communications Manager users in a specific cluster to make and end calls, as well as to access their phone and line configuration.

Cisco Web Dialer applications interact with the Web Dialer servlet through two interfaces:

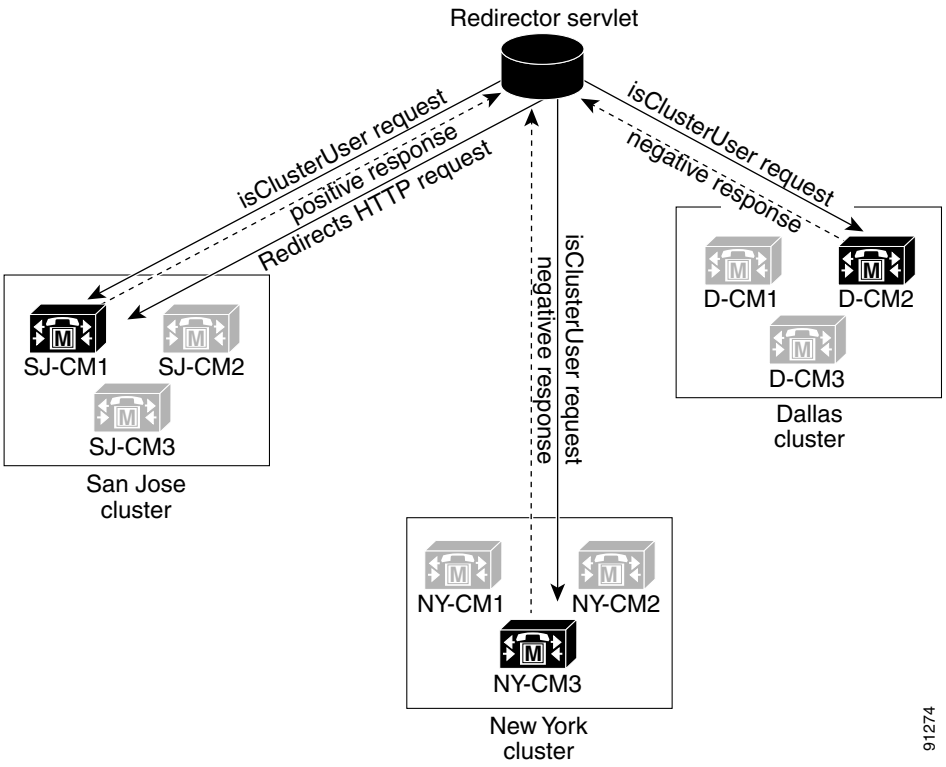
- SOAP over HTTPS—This interface, based on the Simple Object Access Protocol (SOAP), is used to develop desktop applications such as a Microsoft Outlook Plug-in or a SameTime Client Plug-in. Developers can use the isClusterUserSoap interface to design multicluster applications that require functionality similar to a Redirector servlet.
- HTML over HTTPS—This interface, based on the HTTPS protocol, is used to develop web-based applications such as the Unified CM directory search page (directory.asp). Developers who use this interface can use the Redirector servlet for designing multicluster applications.

Redirector Servlet

The Java-based Redirector servlet is responsible for distributing web (HTTP and HTTPS) MakeCall requests to the home Web Dialer server of a user. Redirector generally is used in a multi-cluster environments to instruct an application where to send MakeCall requests. When Redirector receives a MakeCall request, it sends the IsClusterUser broadcast message to all configured Web Dialer servers in the Enterprise. When Redirector receives a positive response, it forwards the request to the appropriate Web Dialer server. Redirector is available for HTTP and HTTPS applications only. SOAP-based applications are responsible for sending the MakeCall request to the home Web Dialer server of a user.

Figure 8-1 illustrates how a Redirector servlet redirects a call in a multicluster environment.

Figure 8-1 Multiple Clusters



Example of Web Dialer Using the Redirector Servlet

For example, consider three clusters, each one in a single city such as San Jose, Dallas, and New York. Each cluster contains three Cisco Unified Communications Manager servers with Web Dialer servlets that have been configured for Cisco Unified Communications Manager servers SJ-CM1, D-CM2, and NY-CM3.

The system administrator configures the Web Dialer servlets on any Cisco Unified Communications Manager server by entering the IP address of that specific Cisco Unified Communications Manager server in the List of WebDialers service parameter.

For information about configuring Web Dialer and Redirector servlets, refer to the "Web Dialer" chapter in the *Cisco Unified Communications Manager Features and Services Guide, Release* 5.0.

When a user who is located in San Jose clicks a telephone number in the corporate directory search page that is enabled by Web Dialer, the following actions happen:

1. The Cisco Unified Communications Manager server sends an initial *makeCall* HTTPS request to the Redirector servlet.
2. If this request is received for the first time, the Redirector servlet reads the Web Dialer server cookie and finds it empty.

   For a repeat request, the Redirector servlet reads the IP address of the Web Dialer server that previously serviced the client and sends a *isClusterUser* HTTPS request only to that server.
3. The Redirector servlet sends back a response that asks for information, which results in the authentication dialog box opening for the user.
4. The user enters the Cisco Unified Communications Manager user ID and password and clicks the **Submit** button.
5. The Redirector servlet reads only the user identification from this information and sends a *isClusterUser* HTTPS request to each Web Dialer server that the system administrator configured.

   Figure 8-1 illustrates how this request is sent to the Web Dialer servlets that have been configured for SJ-CM1, D-CM2, and NY-CM3. Depending on the geographical location of the calling party, the Web Dialer servlet from that cluster responds positively to the Redirector servlet. The remaining Web Dialer servlets that were contacted return a negative response. The Web Dialer servlet SJ-CM1 responds positively to the request because the calling party is located in San Jose (SJ-CM).

The Redirector servlet redirects the original request from the user to SJ-CM1 and sets a cookie on the user browser for future use.

Cisco Web Dialer Security Support

Web Dialer supports secure connections to CTI (TLS connection). For this feature, Web Dialer uses the security API that JTAPI provides. Refer to *Unified CM JTAPI Developers Guide* for the JTAPI API. Web Dialer uses the Application User, "WDSecureSysUser", for obtaining the CTI connection.

You must complete the following configuration before Web Dialer can be configured to open a CTI connection in secure mode.

Step 1 Activate the Cisco CTL Provider service in Cisco Unified Communications Manager Service Administration.

Step 2 Activate the Cisco Certificate Authority Proxy Function Service.

Step 3 Download the Cisco CTL Client from the Application plug-in and install it on any machine.

Step 4 Run the CTL Client, choose the option to "enable Cluster Security," and follow the instructions that display. This requires USB E-tokens.

Step 5 To verify that cluster security is enabled, go to Cisco Unified Communications Manager Administration and look at [System-> Enterprise Parameter configuration]. Look at the Security Parameters; the cluster security should be set to 1.

Step 6 In Cisco Unified Communications Manager Administration, from the User Management drop-down menu, select the Application User CAPF Profile option.

Step 7 Click **Add new InstanceID**.

Step 8 In the CAPF Profile configuration window, set up an InstanceID and CAPF profile for the InstanceID for the Application User WDSecureSysUser.

a. **InstanceID**: Enter the value of instance ID; for example, 001.

b. **Certificate Operation**: Select Install/Upgrade from the drop-down menu.

c. **Authentication Mode**: Select By Authorization String from the drop-down menu.

d. **Authorization String**: Enter the value of authorization string; for example, 12345.

e. **Key Size**: Select key size from drop-down menu; for example, 1024.

f. **Operation Completes By**: Enter the date and time in following format yyyy:mm:dd:hh:mn where yyyy=year, mm=month, dd=date, hh=hour, mn=minutes, such as 2006:07:30:12:30.

Note If this date and time has passed, the certificate update operation will fail.

g. Ignore the **Packet Capture Mode**, **Packet Capture Duration**, and **Certificate** fields.

h. **Certificate Status**: Select Operation pending from the drop-down menu.

If anything else is selected, the certificate update will fail.

Security Service Parameters

Web Dialer includes two mode-specific service parameters for CTI connection security.

- **CTI Manager Connection Security Flag**—This required service parameter indicates whether security for the Web Dialer service CTI Manager connection is enabled or disabled.

  If enabled (true), Cisco Web Dialer will open a secure connection to CTI Manager by using the Application CAPF profile that is configured for the instance ID (as configured in CTI Manager Connection Instance ID service parameter) for Application user WDSecureSysUser. The default value specifies false.

- **Application CAPF Profile Instance ID**: This service parameter specifies the Instance ID of the Application CAPF Profile for Application User WDSecureSysUser that this Web Dialer server will use to open a secure connection to CTI Manager. You must configure this parameter if the CTI Manager Connection Security Flag parameter is enabled (true).

- **Algorithm**:

  1. Read the service parameters.
  2. Get the node IP/name of the nodes where TFTP and CAPF are activated.
  3. For the instanceID (input in service parameters), if the Certificate Operation is 'Install/Upgrade' or 'Delete', delete the current certificates, if any.

4. If the Certificate Operation is not ‘Install/Upgrade’ or ‘Delete’, and a current certificate exists, use this certificate.
5. If no certificate is present, request one by using JTAPI API setSecurityPropertyForInstance; this will need username, instanceID, authCode, tftpServerName, tftpPort, capfServerName, capfPort, certPath, and securityFlag. This call will contact the TFTP server, download the certificate, contact the CAPF server, verify the CTL file, and request the client and server certificates.
6. If Step 5 is successful, set the following items on the ICCNProvider and call open().provider.setInstanceID(instanceID);provider.setTFTPServer(tftpServerName);provider.setCAPFServer(capfServerName);provider.setCertificatePath(certPath);provider.setSecurityOptions(securityFlag);
7. If Step 5 fails, throw initFailedException. You can see this in the Web Dialer traces.

Maximum Concurrent Call Requests

The maximum concurrent call request parameter specifies the maximum number of concurrent Web Dialer call requests per second supported by the Web Dialer service. This value was previously hard-coded. The minimum and maximum values for this throttle are one and eight, and the recommended values for 7825 and 7845 servers are three and six respectively. Version 8.6(1) increased the maximum from six to eight.

For example:

- MCS 7825H2 supports a maximum of two calls per second. Cisco recommends setting the MaxConcurrentCallRequests value to three to allow callers to disconnect the call as needed.
- MCS 7845H2 supports a maximum of four calls per second. Cisco recommends setting the MaxConcurrentCallRequests value to six to allow callers to disconnect the call as needed.

Enter a lower value if RTMT alerts, alarms, or performance counters suggest that the hardware associated with Web Dialer is over-utilized (for example, high CPU and/or memory usage). Enter a higher value to allow more simultaneous WebDialer call requests. Higher values can add more load to the CPU. The default value for this parameter is three.

Phone Support For Cisco Web Dialer

Web Dialer relies on Cisco JTAPI to place calls on the behalf of users. Table 8-2 provides information about CTI supported devices.

Table legend:

- —Supported
- —Not supported

Table 8-2 CTI Supported Device Matrix

Device/Phone Model	SCCP	SIP	Comments
Analog Phone			You can find information on the limitations of this device in *Cisco JTAPI Developer Guide for Cisco Unified CallManager 4.1(3)* .
Cisco 6901			

Table 8-2 CTI Supported Device Matrix (continued)

Device/Phone Model	SCCP	SIP	Comments
Cisco 6911	✓	✓	
Cisco 6921	✓	✓	
Cisco 6941	✓	✓	
Cisco 6961	✓	✓	
Cisco 7902	✓	✗	End of Software Maintenance Release 2007
Cisco 7905	✓	✗	End of Software Maintenance Release 2007
Cisco 7906	✓	✓	
Cisco 7910	✓	✗	End of Software Maintenance Release 2007
Cisco 7911	✓	✓	
Cisco 7912	✓	✗	End of Software Maintenance Release 2007
Cisco 7914 Sidecar	✓	✗	End of Software Maintenance Release 2010
Cisco 7915 Sidecar	✓	✓	
Cisco 7916 Sidecar	✓	✓	
Cisco CKEM Sidecar	✗	✓	
Cisco 7920	✓	✗	End of Software Maintenance Release 2008
Cisco 7921	✓	✗	
Cisco 7925 & 7925-EX	✓	✗	
Cisco 7931	✓	✗	CTI supported only if rollover is disabled. Starting withrelease 7.1 this device is supported when corresponding role is added to user.
Cisco 7935	✓	✗	End of Software Maintenance Release 2005
Cisco 7936	✓	✗	End of Software Maintenance Release 2011
Cisco 7937	✓	✗	
Cisco 7940	✓	✗	End of Software Maintenance Release 2011
Cisco 7941	✓	✓	
Cisco 7941G-GE	✓	✓	End of Software Maintenance Release 2009
Cisco 7942	✓	✓	
Cisco 7945	✓	✓	
Cisco 7960	✓	✗	End of Software Maintenance Release 2011

Table 8-2 CTI Supported Device Matrix (continued)

Device/Phone Model	SCCP	SIP	Comments
Cisco 7961	✓	✓	
Cisco 7961G-GE	✓	✓	End of Software Maintenance Release 2009
Cisco 7962	✓	✓	
Cisco 7965	✓	✓	
Cisco 7970	✓	✓	End of Software Maintenance Release 2009
Cisco 7971	✓	✓	End of Software Maintenance Release 2009
Cisco 7975	✓	✓	
Cisco 7985	✓	✗	End of Software Maintenance Release 2011
Cisco 8941	✓	✗	
Cisco 8945	✓	✗	
Cisco 8961	✗	✓	phoneSetDisplay() interface is not supported
Cisco 9951	✗	✓	phoneSetDisplay() interface is not supported
Cisco 9971	✗	✓	phoneSetDisplay() interface is not supported
Cisco ATA 186	✓	✗	You can find information on the limitations of this device in *Cisco JTAPI Developer Guide for Cisco Unified CallManager 4.1(3)* .
Cisco Cius	✗	✓	CTI support added in release 8.5(1) phoneSetDisplay() interface is not supported XSI interface is not supported. Silent Monitoring/Recording is not supported
Cisco IP Communicator	✓	✓	CTI support added in release 7.1(2)
Cisco Unified Personal Communicator	✗	✗	CTI support when running in desktop mode depends on physical device. CTI support added in release 7.5(1)
Cisco Unified Personal Communicator - Remote Desktop Control Mode	—	—	Refer to the device model under remote control to determine CTI support. Click-to-Answer requires device speakerphone support.
Cisco Unified Communicator Integration for Microsoft Office Communicator/Lync - Softphone Mode	✗	✓	CTI support added in release 8.5(2)

Table 8-2 CTI Supported Device Matrix (continued)

Device/Phone Model	SCCP	SIP	Comments
Cisco Unified Communicator Integration for Microsoft Office Communicator/Lync - Remote Desktop Control Mode	—	—	Refer to the device model under remote control to determine CTI support. Click-to-Answer requires device speakerphone support.
Cisco Web Communicator for Quad - Softphone Mode	—	—	Not a CTI supported device.
Cisco Web Communicator for Quad - Remote Desktop Control Mode	—	—	Refer to the device model under remote control to determine CTI support. Click-to-Answer requires device speakerphone support.
Cisco Unified Communications Integration for WebEx Connect - Softphone Mode	—	—	Not a CTI supported device.
Cisco Unified Communications Integration for WebEx Connect - Remote Desktop Control Mode	—	—	Refer to the device model under remote control to determine CTI support. Click-to-Answer requires device speakerphone support.
Cisco VGC Phone	✓	✗	
VG224	—	—	Not a CTI supported device.
VG248	✓	✗	You can find information on the limitations of this device in *Cisco JTAPI Developer Guide for Cisco Unified CallManager 4.1(3)* .
CTI Port	—	—	CTI supported virtual device that does not use SCCP or SIP
CTI Route Point	—	—	CTI supported virtual device that does not use SCCP or SIP
CTI Route Point (Pilot Point)	—	—	CTI supported virtual device that does not use SCCP or SIP
ISDN BRI Phone	—	—	Not a CTI supported device

Call Flows

The call flows in this section describe the flow of events for client and browser-based applications that use Web Dialer, which should help you design customized applications for Web Dialer.

Desktop-based Client Application Call Flow

Figure 8-2 shows the call flow for an outgoing call from a client application. The user clicks the **Dial** or **Make Call** button in the client application. If the user is making a call for the first time, the application does not have authentication or configuration information on the user.

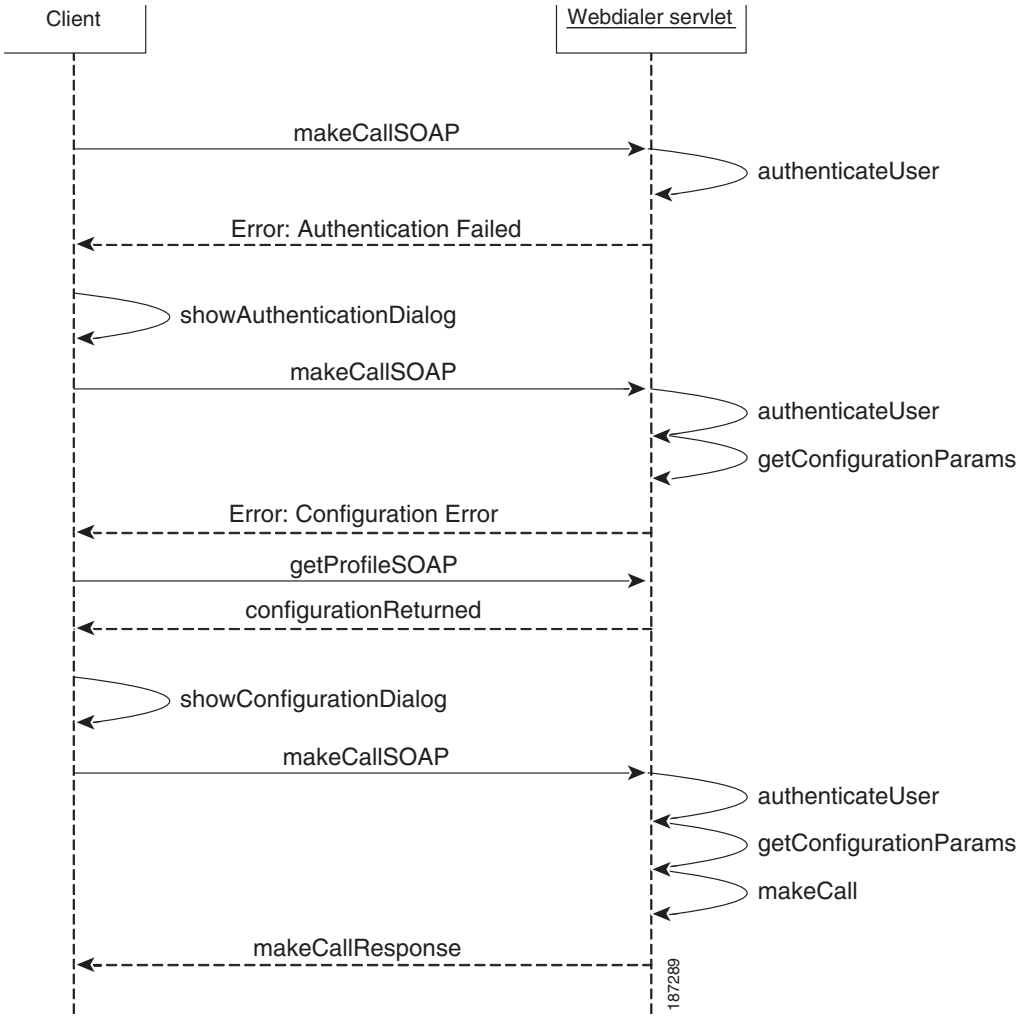
When the user makes a call for the first time,

1. The client sends a makeCallSoap request to the configured Web Dialer servlet.
2. The Web Dialer servlet attempts to authenticate the user. Figure 8-2 shows an authentication failure that occurred because the authentication information is incomplete or does not exist.
3. The Web Dialer servlet sends an authentication failure response to the client application.
4. The client application displays a dialog box that asks for the user ID and password. The user enters this information and clicks the **submit** button. The user ID and password get stored for future invocations of the application.
5. The application sends a repeat SOAP request to the Web Dialer servlet. The request contains credential information on the user.
6. The Web Dialer servlet authenticates the user.
7. The Web Dialer servlet reads any missing configuration information in the request.
8. The Web Dialer servlet returns a configuration error message to the client application.
9. The client application sends a getProfileSoap request to the Web Dialer servlet.
10. The Web Dialer servlet responds with the user configuration information that is stored in the directory.
11. The client application displays a configuration dialog box that asks the user to select or update the configuration. The user enters the information and clicks the **submit** button. The user configuration information gets stored for future invocations of the application.
12. The client resends the makeCallSoap request to the Web Dialer servlet. This request contains the user configuration information.
13. The Web Dialer servlet authenticates the user and dials the telephone number by using the information that the makeCallSoap request contains. It responds to the client with a success or failure message.

Note The call flow goes directly to step 12 in these situations:

- If the credential and configuration information is already stored when the application is installed.
- For all subsequent requests that the user makes.

Figure 8-2 Cisco Web Dialer Call Flow for a Client-Based Application

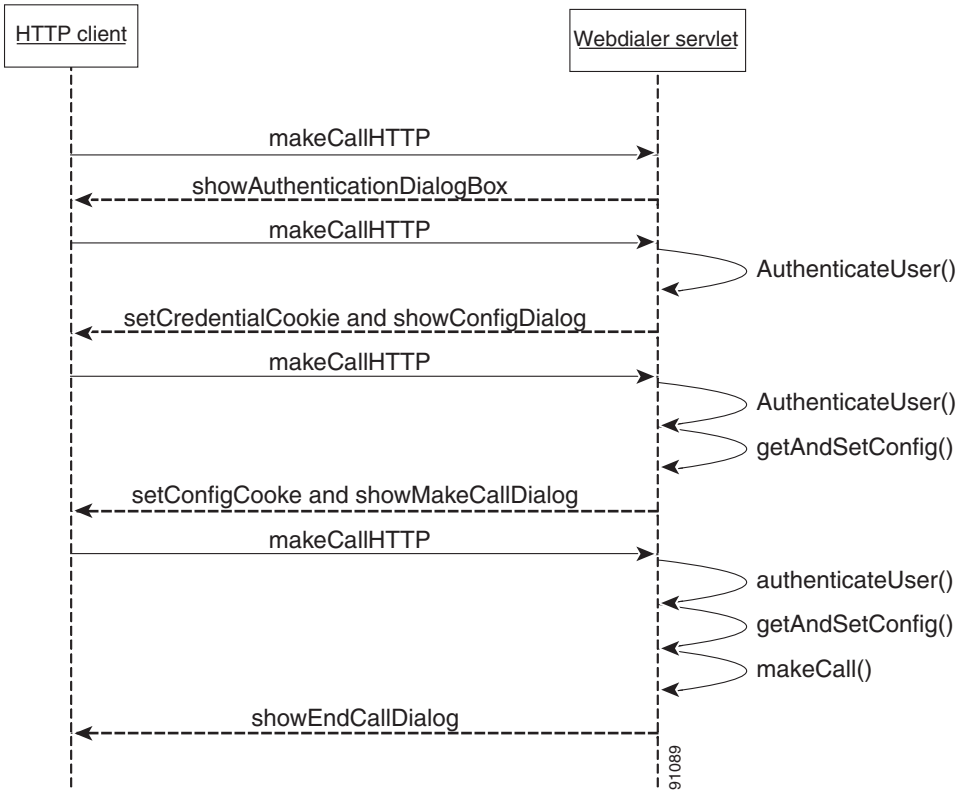


Browser-Based Application Call Flow

Figure 8-3 shows the call flow for an HTTP-based browser application such as a directory search page, personal address book, or the Cisco Unified Communications Manager directory search page (directory.asp).

The user clicks the **Dial** or **Make Call** button in the address book of the client application. If the user is making a call for the first time, the application does not have authentication or configuration information on the user.

Figure 8-3 Cisco Web Dialer Call Flow for a Browser-Based Application



When the user makes a call for the first time:

1. The client sends a makeCall HTTPS request to the configured Web Dialer servlet. The query string contains the number to be called.
2. The Web Dialer servlet authenticates the user. Authentication fails because the authentication information is incomplete or does not exist.

Note Authentication succeeds if the user credentials are sent with the request, and the call flow goes directly to step 7.

3. The Web Dialer servlet sends an authentication dialog to the client browser for user authentication.
4. The user enters the user ID and password and clicks the **Submit** button.
5. The client sends a makeCallHTTPS request that contains the user credentials to the Web Dialer servlet.
6. The Web Dialer servlet authenticates the user.
7. The Web Dialer servlet reads the configuration information in the cookie that is sent with the request.
8. Assuming that the request is made for the first time, the servlet sends a response that contains a cookie to the client browser. The cookie that contains the client credentials gets stored on the client browser. The client credentials comprise user ID, IP address, and the time of the request.
9. The user enters the updates in the configuration dialog box and clicks the **Submit** button.
10. The client browser sends a makeCall HTTPS request to the Web Dialer servlet. The request contains a cookie with the credential and configuration information in parameter form.

11. The Web Dialer servlet uses the credentials to authenticate the user and saves the configuration information in its memory.
12. The Web Dialer servlet sends a makeCall confirmation dialog to the client browser with the configuration information that is stored in a cookie. The cookie gets stored on the client browser for future invocations.
13. The Make Call dialog box appears on the user computer screen. The user clicks the **Dial** button, which sends another makeCall HTTPS request to the Web Dialer servlet.
14. The Web Dialer servlet authenticates the user by using the credentials in the cookie, retrieves the configuration information from the cookie, and makes the call.
15. The servlet responds by sending an endCall confirmation dialog to the user to end the call. The End Call dialog box appears on the user computer screen and stays there for the time interval that is configured in the service parameters.

For subsequent requests, the call flow starts at step 12 and ends at step 15.

Interfaces

Web Dialer applications interact with the Web Dialer servlet through two interfaces:

- SOAP over HTTPS—This interface, based on the Simple Object Access Protocol (SOAP), is used to develop desktop applications such as a Microsoft Outlook Plug-in and SameTime Client Plug-in. Developers can use the isClusterUserSoap interface to design multicluster applications that require functionality similar to a Redirector servlet.
- HTML over HTTPS—This interface, based on the HTTPS protocol, is used to develop web-based applications such as the Cisco Unified Communications Manager directory search page (directory.asp). Developers who are using this interface can use the Redirector servlet for designing multicluster applications.

SOAP Over HTTPS Interface

To access the SOAP interfaces for Web Dialer, use the Web Dialer Web Service Definition Language (WSDL) in the "Cisco Web Dialer WSDL" section on page 8-26.

makeCallSoap

You access the makeCallSoap interface by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Parameter	Mandatory	Description	Data Type	Range Values	Default Value
Destination	Mandatory	Standard canonical form. For example, +1 408 5551212 or extensions such as 2222. The optional service parameter "Apply Application Dial Rules on SOAP Dial Request" is False by default; if this parameter is True, the destination gets transformed according to the dial rules.	String	None	None
Credential	Mandatory	The user ID or password of the user or proxy user. For more information on creating a proxy user, see the *Cisco Web Dialer* chapter in the *Cisco Unified Communications Manager Features and Services Guide, Release 5.0.*	Refer to the credential data type in the "Cisco Web Dialer WSDL" section on page 8-26.	None	None
Profile	Mandatory	The profile that is used to make a call. A typical profile is a calling device such as an IP phone or line. The line should be in the same format as returned by getProfileSoap—<number>; <partition>	Refer to the profile data type in the "Cisco Web Dialer WSDL" section on page 8-26.	None	None

Results

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Error Code	Name	Type	Description	Action by application
0	responseCode	Integer	Success	Displays a dialog box.
	responseDescription	String	Success	
1	responseCode	Integer	Call failure error	Displays a relevant error message.
	responseDescription	String	Call failure error	
2	responseCode	Integer	Authentication error	Displays the authentication dialog where the user enters ID and password information.
	responseDescription	String	User authentication error	

Error Code	Name	Type	Description	Action by application
3	responseCode	Integer	No authentication proxy rights	Void for user-based applications.
	responseDescription	String	No authentication proxy rights	
4	responseCode	Integer	Directory error	Displays an appropriate directory error message.
	responseDescription	String	Directory error	
5	responseCode	Integer	No device is configured for the user, or missing parameters exist in the request.	The application initiates a getProfileSoap request and displays the selected device and line to the user.
	responseDescription	String	No device is configured for the user, or missing parameters exist in the request.	
6	responseCode	Integer	Service temporarily unavailable	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service temporarily unavailable	
7	responseCode	Integer	Destination cannot be reached.	Displays the appropriate error dialog that allows the user to edit the dialed number.
	responseDescription	String	Destination cannot be reached.	
8	responseCode	Integer	Service error	Displays the appropriate error dialog.
	responseDescription	String	Service error	
9	responseCode	Integer	Service overloaded	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service overloaded	

This example shows a makeCallSoap request:

```
<?xml version="1.0" encoding="utf-8" ?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:makeCallSoap soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="urn:Credential">
            <userID xsi:type="xsd:string">wd</userID>
            <password xsi:type="xsd:string">55555</password>
         </in0>
         <in1 xsi:type="soapenc:string"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">1234</in1>
         <in2 xsi:type="urn:UserProfile">
            <user xsi:type="xsd:string">wd</user>
```

```
            <deviceName xsi:type="xsd:string">SEP001558C8970F</deviceName>
            <lineNumber xsi:type="xsd:string">1234</lineNumber>
            <supportEM xsi:type="xsd:boolean">false</supportEM>
            <locale xsi:type="xsd:string">English</locale>
            <dontAutoClose xsi:type="xsd:boolean">false</dontAutoClose>
            <dontShowCallConf xsi:type="xsd:boolean">true</dontShowCallConf>
         </in2>
      </urn:makeCallSoap>
   </soapenv:Body>
</soapenv:Envelope>
```

endCallSoap

You access the endCallSoap interface by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Parameter	Mandatory	Description	Data Type	Range Values	Default Value
Credential	Mandatory	The user ID or password of the user or proxy user. For information on creating a proxy user, see the *Cisco Web Dialer* chapter in the *Cisco Unified Communications Manager Features and Services Guide, Release 5.0*.	Refer to the credential data type in "Cisco Web Dialer WSDL" section on page 8-26.	None	None
Profile	Mandatory	The profile that is used to make a call. A typical profile is a calling device such as an IP phone or line.	Refer to the profile data type in the "Cisco Web Dialer WSDL" section on page 8-26.	None	None

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Error Code	Name	Type	Description	Action by application
0	responseCode	Integer	Success	Displays a dialog box on the computer screen.
	responseDescription	String	Success	
1	responseCode	Integer	Call failure error	Displays a relevant error message.
	responseDescription	String	Call failure error	
2	responseCode	Integer	Authentication error	Displays authentication dialog for user to enter user ID and password.
	responseDescription	String	User authentication error	
3	responseCode	Integer	No authentication proxy rights	Void for user-based applications.
	responseDescription	String	No authentication proxy rights	

Error Code	Name	Type	Description	Action by application
4	responseCode	Integer	Directory error	Displays an appropriate directory error message.
	responseDescription	String	Directory error	
5	responseCode	Integer	No device is configured for the user, or missing parameters exist in the request.	The Application initiates a getProfileSoap request and displays the selected device and line to the user.
	responseDescription	String	No device is configured for the user, or missing parameters exist in the request.	
6	responseCode	Integer	Service temporarily unavailable	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service temporarily unavailable	
7	responseCode	Integer	Destination cannot be reached.	Displays the appropriate error dialog that allows the user to edit the dialed number.
	responseDescription	String	Destination cannot be reached.	
8	responseCode	Integer	Service error	Displays appropriate error dialog.
	responseDescription	String	Service error	
9	responseCode	Integer	Service overloaded	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service overloaded	

This example shows an endCallSoap request:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:endCallSoap soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="urn:Credential">
            <userID xsi:type="xsd:string">wd</userID>
            <password xsi:type="xsd:string">55555</password>
         </in0>
         <in1 xsi:type="urn:UserProfile">
            <user xsi:type="xsd:string">wd</user>
            <deviceName xsi:type="xsd:string">SEP001558C8970F</deviceName>
            <lineNumber xsi:type="xsd:string">1234</lineNumber>
            <supportEM xsi:type="xsd:boolean">false</supportEM>
            <locale xsi:type="xsd:string">English</locale>
            <dontAutoClose xsi:type="xsd:boolean">false</dontAutoClose>
            <dontShowCallConf xsi:type="xsd:boolean">true</dontShowCallConf>
         </in1>
      </urn:endCallSoap>
   </soapenv:Body>
```

```
</soapenv:Envelope>
```

getProfileSoap

You access the getProfileSoap interface, which is used by plug-in based clients, by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Parameter	Mandatory/ Optional	Description	Data Type	Value Range	Default Value
Credential	Mandatory	User ID or password of the user or proxy user. For information on creating a proxy user, see the *Cisco Web Dialer* chapter in *Cisco Unified Communications Manager Features and Services Guide, Release 5.0*.	Refer to the credential data type in the "Cisco Web Dialer WSDL" section on page 8-26.	None	None
UserID	Mandatory	The user ID for which the configuration is requested.	String	None	None

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Error Code	Name	Type	Description	Action by plug-in application
0	responseCode	Integer	Returns an array of phones or lines on the phone that is associated with the user. Refer to the Cisco Web Dialer WSDL for the WDDeviceInfo data type. **Note** getProfileSoap API does not return the Extension Mobility device profiles associated with the user.	Displays a dialog box on the computer screen.
	responseDescription	String	Success	
	deviceInfoList	Array	Returns an array of the the WDDeviceInfo data type	
1	responseCode	Integer	No device configured for the user	Displays an appropriate error message.
	responseDescription	String	No device configured for the user	

Error Code	Name	Type	Description	Action by plug-in application
2	responseCode	Integer	Authentication error	Displays the authentication dialog where the user enters ID and password information.
	responseDescription	String	User authentication error	
3	responseCode	Integer	No authentication proxy rights	Void for user-based applications.
	responseDescription	String	No authentication proxy rights	
4	responseCode	Integer	Directory error	Displays an appropriate directory error message.
	responseDescription	String	Directory error	
6	responseCode	Integer	Service temporarily unavailable	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service temporarily unavailable	
9	responseCode	Integer	Service overloaded	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service overloaded	

This example shows a getProfileSoap request used for debugging purposes (normally, the SOAP implementation layer would make this request):

```
<?xml version="1.0" ?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:getProfileSoap
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="urn:Credential">
            <userID xsi:type="xsd:string">wd</userID>
            <password xsi:type="xsd:string">55555</password>
         </in0>
         <in1 xsi:type="soapenc:string"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">wd</in1>
      </urn:getProfileSoap>
   </soapenv:Body>
</soapenv:Envelope>
```

isClusterUserSoap

You access the isClusterUserSoap interface by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Use this SOAP interface for multicluster applications that require functionality, similar to a Redirector servlet, for redirecting calls to the various locations where Web Dialer is installed on a network. The application uses this interface to locate and verify the Web Dialer that is servicing the user, followed by makeCall, endCall, or getProfile requests to that Web Dialer.

Parameter	Mandatory	Description	Data Type	Range of Values	Default Value
UserID	Mandatory	The user ID for which the the request is made.	String	None	None

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Name	Type	Description
result	Boolean	The result specifies True if the user is present in the directory of the cluster. The result specifies False if the user is not present in the directory.

This example shows an isClusterUserSoap request:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:isClusterUserSoap
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="soapenc:string"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">wd</in0>
      </urn:isClusterUserSoap>
   </soapenv:Body>
</soapenv:Envelope>
```

getProfileDetailSoap

You access the getProfileDetailSoap interface by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Parameter	Mandatory	Description	Data Type	Range Values	Default Value
Credential	Mandatory	User ID or password of the user or proxy user. For information about creating a proxy user, refer to the "Cisco Web Dialer" chapter in *Cisco Unified Communications Manager Features and Services Guide, Release 5.0*.	See the credential data type in the "Cisco Web Dialer WSDL" section on page 8-26.	None	None

Results

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Error Code	Name	Type	Description	Action by application
0	responseCode	Integer	Returns an array of phones or lines on the phone that is associated with the user. Also returns Phone Description and the Phone type for each device. See the credential data type in the "Cisco Web Dialer WSDL" section on page 8-26.	Displays a dialog box.
	responseDescription	String	Success	
	deviceInfoListDetail	Array	Returns an array of the the WDDeviceInfoDetail data type	
1	responseCode	Integer	No device configured for the user	Displays an appropriate error message.
	responseDescription	String	No device configured for the user	
2	responseCode	Integer	Authentication error	Displays the authentication dialog where the user enters ID and password information.
	responseDescription	String	User authentication error	

Error Code	Name	Type	Description	Action by application
3	responseCode	Integer	No authentication proxy rights	Void for user-based applications.
	responseDescription	String	No authentication proxy rights	
4	responseCode	Integer	Directory error	Displays an appropriate directory error message.
	responseDescription	String	Directory error	
6	responseCode	Integer	Service temporarily unavailable	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service temporarily unavailable	
9	responseCode	Integer	Service overloaded	Displays the appropriate error dialog with an option to try again.
	responseDescription	String	Service overloaded	

This example shows a getProfileDetailSoap request used for debugging purposes (normally, the SOAP implementation layer would make this request):

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:getProfileDetailSoap
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="urn:Credential">
            <userID xsi:type="xsd:string">wd</userID>
            <password xsi:type="xsd:string">55555</password>
         </in0>
      </urn:getProfileDetailSoap>
   </soapenv:Body>
</soapenv:Envelope>
```

getPrimaryLine

You access the getPrimaryLine interface by initiating a SOAP request to the URL https://<CUCM_Server>/webdialer/services/WebdialerSoapService70 where CUCM_Server specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Parameter	Mandatory	Description	Data Type	Range of Values	Default Value
UserID	Mandatory	The user ID for which the the request is made.	String	None	None

See the "Cisco Web Dialer WSDL" section on page 8-26 for return values and their data type.

Name	Type	Description
result	Boolean	The result returns the number that is configured by the Unified CM administrator as the primary line of the user.

This example shows a getPrimaryLine request:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:WD70">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:getPrimaryLine
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <in0 xsi:type="urn:Credential">
            <userID xsi:type="xsd:string">wd</userID>
            <password xsi:type="xsd:string">55555</password>
         </in0>
      </urn:getPrimaryLine>
   </soapenv:Body>
</soapenv:Envelope>
```

HTML Over HTTPS Interfaces

This section describes the HTML over HTTPS interfaces.

Note If you are using the browser interface, then use the HTTP POST method to pass the parameters. This reduces the time delay that the Web Dialer takes to automatically convert GET parameters to POST.

makeCall

You use the makeCall interface in customized directory search applications. The Unified CM directory search page (directory.asp) also uses this interface. Access the makeCall interface by initiating an HTTPS request to the URL https://<ipaddress>/webdialer/Webdialer. In this URL, ipaddress specifies the IP address of the Cisco Unified Communications Manager server where Web Dialer is configured.

Browser-based applications in which the browser accepts cookies use this interface. The user profile exists only for the length of the session if the cookies are disabled in a browser. For a sample script that is used to enable directory search pages, go to the "Sample Code Snippet" section on page 8-30.

Parameter	Mandatory	Description	Data Type	Range of Values	Default Value
destination	Mandatory	Destination number called by the application. Number gets converted to a regular telephone number by applying the application dial rules. Refer to the *Cisco Web Dialer* chapter in the *Cisco Unified Communications Manager Features and Services Guide, Release 5.0*.	String	None	None

Name	Description
result	Web Dialer displays the appropriate dialog and its applicable success or error message. It displays an authentication dialog if no active session exists.

makeCallProxy

You access the makeCallProxy interface by initiating an HTTPS request to the URL https://ipaddress/webdialer/Webdialer?cmd=doMakeCallProxy. Browser-based applications in which the browser accepts cookies use this interface. If the cookies are disabled in a browser, the user profile exists only for the length of the session.

Applications such as a personal address book, defined in the Unified CMUser pages at https://cmserver/CMUser, can use the makeCallProxy interface. The credential of the application is used as a proxy to make calls on behalf of users. Because these users have authenticated themselves before accessing the Unified CMUser window, they do not get prompted again for their user ID and password. The application sends the user ID and password of the proxy user in the form of a query string in the request or as a parameter in the body of the POST message.

Note The API does not use the M-POST method as defined in the HTTP Extension Framework.

For a sample script that is used to enable directory search pages, go to the "Sample Code Snippet" section on page 8-30.

Parameter	Mandatory	Description	Data Type	Range of Values	Default Value
uid	Mandatory	The user ID for which the request is made	String	None	None
appid	Mandatory	The userid of the application that is making a request on behalf of the user. For example, consider a Unified CM personal address book where the application allows authentication proxy rights. The appid parameter is used when the user logs in once; for example, in the Unified CM User windows. After this login, other pages do not require the user to log in again. For web page applications that are not integrated, the appid matches the userid.	String	None	None
pwd	Mandatory	The password of the appid	String	None	None
destination	Mandatory	The number to be called. The dial plan service converts this number to an E.164 number.	String	None	None

Name	Description
result	Web Dialer displays the appropriate dialog and its applicable success or error message.

Cisco Web Dialer WSDL

The WSDL specification provides the basis for the Web Service Definition Language (WSDL) for Web Dialer. You can access the WSDL for Web Dialer on the Web Dialer server installation at

https://<CCM_Server>/webdialer/wsdl/wd70.wsdl

Use this specific WSDL and the interfaces that are mentioned in this document to develop customized applications for Web Dialer. For a list of references on Cisco Unified Communications Manager, SOAP, and WSDL, refer to the Related Documentation section.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:WD70"
xmlns:apachesoap="http://xml.apache.org/xml-soap" xmlns:impl="urn:WD70"
xmlns:intf="urn:WD70" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<!--WSDL created by Apache Axis version: 1.4 With AXIS-2250
Built on Apr 22, 2006 (06:55:48 PDT)-->
 <wsdl:types>
  <schema targetNamespace="urn:WD70" xmlns="http://www.w3.org/2001/XMLSchema">
   <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
```

```
<complexType name="Credential">
 <sequence>
  <element name="userID" type="xsd:string"/>
  <element name="password" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="UserProfile">
 <sequence>
  <element name="user" type="xsd:string"/>
  <element name="deviceName" type="xsd:string"/>
  <element name="lineNumber" type="xsd:string"/>
  <element name="supportEM" type="xsd:boolean"/>
  <element name="locale" type="xsd:string"/>
  <element name="dontAutoClose" type="xsd:boolean"/>
  <element name="dontShowCallConf" type="xsd:boolean"/>
 </sequence>
</complexType>
<complexType name="CallResponse">
 <sequence>
  <element name="responseCode" type="xsd:int"/>
  <element name="responseDescription" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="WDDeviceInfo">
 <sequence>
  <element name="deviceName" type="xsd:string"/>
  <element name="lines" type="impl:ArrayOf_soapenc_string"/>
 </sequence>
</complexType>
<complexType name="ArrayOfWDDeviceInfo">
 <complexContent>
  <restriction base="soapenc:Array">
   <attribute ref="soapenc:arrayType" wsdl:arrayType="impl:WDDeviceInfo[]"/>
  </restriction>
 </complexContent>
</complexType>
<complexType name="GetConfigResponse">
 <sequence>
  <element name="description" type="xsd:string"/>
  <element name="deviceInfoList" type="impl:ArrayOfWDDeviceInfo"/>
  <element name="responseCode" type="xsd:int"/>
 </sequence>
</complexType>
<complexType name="ArrayOf_soapenc_string">
 <complexContent>
  <restriction base="soapenc:Array">
   <attribute ref="soapenc:arrayType" wsdl:arrayType="soapenc:string[]"/>
  </restriction>
 </complexContent>
</complexType>
<complexType name="WDDeviceInfoDetail">
 <sequence>
  <element name="deviceName" nillable="true" type="soapenc:string"/>
  <element name="lines" nillable="true" type="impl:ArrayOf_soapenc_string"/>
  <element name="phoneDesc" nillable="true" type="soapenc:string"/>
  <element name="phoneType" nillable="true" type="soapenc:string"/>
 </sequence>
</complexType>
<complexType name="ArrayOfWDDeviceInfoDetail">
 <complexContent>
  <restriction base="soapenc:Array">
   <attribute ref="soapenc:arrayType" wsdl:arrayType="impl:WDDeviceInfoDetail[]"/>
  </restriction>
 </complexContent>
```

```
    </complexType>
    <complexType name="ConfigResponseDetail">
     <sequence>
      <element name="description" nillable="true" type="soapenc:string"/>
      <element name="deviceInfoListDetail" nillable="true"
type="impl:ArrayOfWDDeviceInfoDetail"/>
      <element name="responseCode" type="xsd:int"/>
     </sequence>
    </complexType>
   </schema>
  </wsdl:types>
   <wsdl:message name="getProfileDetailSoapResponse">
      <wsdl:part name="getProfileDetailSoapReturn" type="impl:ConfigResponseDetail"/>
    </wsdl:message>
    <wsdl:message name="getPrimaryLineResponse">
      <wsdl:part name="getPrimaryLineReturn" type="soapenc:string"/>
    </wsdl:message>
    <wsdl:message name="getPrimaryLineRequest">
      <wsdl:part name="in0" type="impl:Credential"/>
    </wsdl:message>
    <wsdl:message name="getProfileDetailSoapRequest">
      <wsdl:part name="in0" type="impl:Credential"/>
    </wsdl:message>
    <wsdl:message name="getProfileSoapRequest">
      <wsdl:part name="in0" type="impl:Credential"/>
      <wsdl:part name="in1" type="soapenc:string"/>
    </wsdl:message>
    <wsdl:message name="getProfileSoapResponse">
      <wsdl:part name="getProfileSoapReturn" type="impl:GetConfigResponse"/>
    </wsdl:message>
    <wsdl:message name="endCallSoapResponse">
      <wsdl:part name="endCallSoapReturn" type="impl:CallResponse"/>
    </wsdl:message>
    <wsdl:message name="makeCallSoapResponse">
      <wsdl:part name="makeCallSoapReturn" type="impl:CallResponse"/>
    </wsdl:message>
    <wsdl:message name="isClusterUserSoapResponse">
      <wsdl:part name="isClusterUserSoapReturn" type="xsd:boolean"/>
    </wsdl:message>
    <wsdl:message name="makeCallSoapRequest">
      <wsdl:part name="in0" type="impl:Credential"/>
      <wsdl:part name="in1" type="soapenc:string"/>
      <wsdl:part name="in2" type="impl:UserProfile"/>
    </wsdl:message>
    <wsdl:message name="endCallSoapRequest">
      <wsdl:part name="in0" type="impl:Credential"/>
      <wsdl:part name="in1" type="impl:UserProfile"/>
    </wsdl:message>
    <wsdl:message name="isClusterUserSoapRequest">
      <wsdl:part name="in0" type="soapenc:string"/>
    </wsdl:message>
    <wsdl:portType name="WDSoapInterface">
      <wsdl:operation name="makeCallSoap" parameterOrder="in0 in1 in2">
        <wsdl:input message="impl:makeCallSoapRequest" name="makeCallSoapRequest"/>
        <wsdl:output message="impl:makeCallSoapResponse" name="makeCallSoapResponse"/>
      </wsdl:operation>
      <wsdl:operation name="endCallSoap" parameterOrder="in0 in1">
        <wsdl:input message="impl:endCallSoapRequest" name="endCallSoapRequest"/>
        <wsdl:output message="impl:endCallSoapResponse" name="endCallSoapResponse"/>
      </wsdl:operation>
      <wsdl:operation name="getProfileSoap" parameterOrder="in0 in1">
        <wsdl:input message="impl:getProfileSoapRequest" name="getProfileSoapRequest"/>
        <wsdl:output message="impl:getProfileSoapResponse"
name="getProfileSoapResponse"/>
```

```
      </wsdl:operation>
      <wsdl:operation name="isClusterUserSoap" parameterOrder="in0">
         <wsdl:input message="impl:isClusterUserSoapRequest"
name="isClusterUserSoapRequest"/>
         <wsdl:output message="impl:isClusterUserSoapResponse"
name="isClusterUserSoapResponse"/>
      </wsdl:operation>
      <wsdl:operation name="getProfileDetailSoap" parameterOrder="in0">
         <wsdl:input message="impl:getProfileDetailSoapRequest"
name="getProfileDetailSoapRequest"/>
         <wsdl:output message="impl:getProfileDetailSoapResponse"
name="getProfileDetailSoapResponse"/>
      </wsdl:operation>
      <wsdl:operation name="getPrimaryLine" parameterOrder="in0">
         <wsdl:input message="impl:getPrimaryLineRequest" name="getPrimaryLineRequest"/>
         <wsdl:output message="impl:getPrimaryLineResponse"
name="getPrimaryLineResponse"/>
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding name="WebdialerSoapServiceSoapBinding" type="impl:WDSoapInterface">
      <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="makeCallSoap">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="makeCallSoapRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="makeCallSoapResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="endCallSoap">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="endCallSoapRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="endCallSoapResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="getProfileSoap">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="getProfileSoapRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="getProfileSoapResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="isClusterUserSoap">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="isClusterUserSoapRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="isClusterUserSoapResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
```

```
      </wsdl:operation>
      <wsdl:operation name="getProfileDetailSoap">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="getProfileDetailSoapRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="getProfileDetailSoapResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="getPrimaryLine">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="getPrimaryLineRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="getPrimaryLineResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:WD70" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="WDSoapInterfaceService">
      <wsdl:port binding="impl:WebdialerSoapServiceSoapBinding"
name="WebdialerSoapService">
         <wsdlsoap:address
location="https://localhost/webdialer/services/WebdialerSoapService70"/>
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

Sample Code Snippet

This sample code snippet enables Web Dialer from a directory search page.

Single-Cluster Applications

Use this snippet for single-cluster applications if all users are in only one cluster.

```
f<FORM action="https://42.88.86.1/webdialer/Webdialer" method="post">
    <P>
    <INPUT type="hidden" name="destination" value="+666">
    <INPUT type="submit" value="Send">
    </P>
 </FORM>
```

Multiple-Cluster Applications

Use this snippet if all users are spread across different clusters.

```
function launchWebDialerWindow( url ) {
    webdialer=window.open( url, "webdialer", "status=no, width=420, height=300,
scrollbars=no, resizable=yes, toolbar=no" );
  }

  function launchWebDialerServlet( destination ) {
    url= 'https://<%=server_name%>/webdialer/Redirector?destination='+escape(destination);
    launchWebDialerWindow( url );
  }
```

```
!These functions can be called from the HTML page which has a hyperlink to the phone
number to be called. An example of it is
<TD><A href="javascript:launchWebDialerServlet( <%= userInfo.TelephoneNumber %> )"><%=
userInfo.TelephoneNumber %></A> </TD>
```

CHAPTER 9

Cisco Web Dialer Operations By Release

Table 9-1 lists new, changed, and deprecated Cisco Web Dialer operations by release. It also lists operations that are under consideration or review (UCR). Operation details can be found in Chapter 8, "Cisco Web Dialer API Programming."

Note Beginning with Cisco Unified Communications Manager Release 5.1, Web Dialer SOAP operations use HTTPS (SSL). HTTP was used in releases earlier than 5.1.

Table legend:

- —Supported
- —Not supported
- —Modified

Operations By Release

Table 9-1 Web Dialer Operations by Cisco Unified Communications Manager Release

Operation	5.0	5.1	6.0	6.1	7.0	7.1(2)	7.1(3)	8.0(1)	8.5(1)	8.6(1)
endCallSoap	✓	ℹ	✓	✓	✓	✓	✓	✓	✓	✓
getPrimaryLine	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓
getProfileDetailSoap	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓
getProfileSoap	✓	ℹ	✓	✓	✓	✓	✓	✓	✓	✓
isClusterUserSoap	✓	ℹ	✓	✓	✓	✓	✓	✓	✓	✓
makeCallSoap	✓	ℹ	✓	✓	✓	✓	✓	✓	✓	✓

Note Release 5.1 onwards SOAP is over HTTPS (SSL) instead of the earlier HTTP.

PART 5

Routing Rules API

CHAPTER 10

Cisco Unified Routing Rules Interface

This chapter describes the Cisco Unified Routing Rules Interface.

It contains the following sections:

Overview

Cisco Unified Communication Manager (Unified CM) 8.0(1) and later supports the External Call Control (ECC) feature. This feature enables an adjunct route server to make call-routing decisions for Unified CM by using the 8.0(1) and later Cisco Unified Routing Rules interface. On configuring the external call control feature, Unified CM issues a call routing request when the called number matches the external call control enabled pattern. The call routing request contains the calling party and called party information to the adjunct route server. The adjunct route server receives the request, applies appropriate business logic, and returns a call routing response that instructs Unified CM on how the call should be routed, along with any additional call treatment that should be applied.

The adjunct route server can instruct Unified CM to:

- Allow, divert, or deny the call.
- Modify calling and called party information.
- Play announcements to callers.
- Reset call history so that the adjunct voicemail and IVR servers can properly interpret calling or called party information.
- Log reason codes that indicate why calls were diverted or denied.

The following examples show how external call control works:

- Best Quality Voice Routing—The adjunct route server monitors network link availability, bandwidth usage, latency, jitter, MOS scores, and returns routing directives to ensure that calls are routed through voice gateways that deliver the best voice quality to all call participants.
- Least Cost Routing—The adjunct route server is configured with carrier contract information such as Lata and Inter-Lata rate plans, trunking costs, and burst utilization costs to ensure calls are routed over the most cost effective links.
- Ethical Wall—The adjunct route server is configured with corporate policies that determines which users are allowed to communicate with each other and ensures that only permitted communications are allowed.

For information on how to configure External Call Control, refer to the *Cisco Unified Communications Manager Features and Services Guide* for Release 8.0(1) at *http://www.cisco.com/en/US/products/sw/voicesw/ps556/prod_maintenance_guides_list.html*.

New and Changed Information

The following sections provide information on the changes in the Routing Rules APIs in Unified CM release 8.6(1) and the previous releases:

New Information for Cisco Unified Communications Manager 8.6(1)

There are no changes in the Routing Rules API in Unified CM release 8.6(1).

New and Changed Information in Previous Releases of Unified CM

This section provides the new and changed information in previous versions of Unified Communications Manager.

New Information for Cisco Unified Communications Manager 8.5(1)

There are no changes in the Routing Rules API in Unified CM release 8.5(1).

New Information for Cisco Unified Communications Manager 8.0(1)

This is the initial release of the routing rules interface.

Call Routing Request

This section provides information on the call routing request message. The following topics are covered:

Description

Unified CM sends the eXtensible Access Control Markup Language (XACML) based routing request over HTTP or HTTPS using the POST method. The XACML requests comply with XACML v2.0 standard.

Note For more information on XACML, refer to *XACML 2.0 Core: eXtensible Access Control Markup Language (XACML) Version 2.0* document at *http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf*

The XACML request contains the following information about the call:

- The calling and called numbers
- The transformed calling and called numbers, that is, the results of the calling and called number transformation applied at the translation pattern.
- The type of trigger point. TranslationPattern is the only trigger point type that is supported in Unified CM release 8.0(1).

Format

The following is the call routing request format:

```
POST <Request-URI> HTTP/1.1
```

```
Host: <route server>:<port>
Accept: */*
Content-type: text/xml; charset=ISO-8859-1
methodName: <string>
User-Agent: CiscoUCM-HttpClient/1.0
Connection:Keep-Alive
Content-Length: <length>
<?xml encoding=”UTF-8” version=”1.0”?>
<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os">
<Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
    <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role-id"
    DataType="http://www.w3.org/2001/XMLSchema#string" Issuer="requestor"
    <AttributeValue>[string]</AttributeValue> </Attribute>
    <Attribute AttributeId="urn:Cisco:uc:1.0:callingnumber"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>[number]</AttributeValue> </Attribute>
    <Attribute AttributeId="urn:Cisco:uc:1.0:callednumber"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>[number]</AttributeValue> </Attribute>
    <Attribute AttributeId="urn:Cisco:uc:1.0:transformedcgpn"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>[number]</AttributeValue> </Attribute>
    <Attribute AttributeId="urn:Cisco:uc:1.0:transformedcdpn"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>[number]</AttributeValue> </Attribute>
</Subject>
<Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
    DataType="http://www.w3.org/2001/XMLSchema#anyURI">
    <AttributeValue>[string]</AttributeValue> </Attribute>
</Resource>
<Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
    DataType="http://www.w3.org/2001/XMLSchema#anyURI">
    <AttributeValue>[URI]</AttributeValue> </Attribute>
</Action>
<Environment>
    <Attribute AttributeId="urn:cisco:1.0:triggerpointtype"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>[string]</AttributeValue></Attribute>
</Environment>
</Request>
```

Parameters

This section provides the description of various parameters in the request format.

Parameters	Description
HTTP POST Message	
Request-URI	You can provision the Request-URI in an external call control profile. The Cisco Unified CM administrator can customize the URI for a particular web service in a deployment. For example, when an XACML 2.0 compliant policy server provides the web service, the Request-URI can be set to */pdp/AuthorizationEndPoint*
Host:<route server>:<port>	The host and port of the web service to which the request is sent. For example, Host:*cepm1.cisco.com:8080*

Parameters	Description
methodName	The name of the method. For example, *isRoleAccessAllowed* is used when Cisco Enterprise Policy Manager is the Policy Decision Point (PDP).
User-Agent	The User-Agent field is set to *CiscoUCM-HttpClient/1.0*
Content-Length	The Content-Length field indicates the size of the entity-body in octets.
XACML Message	
role-id	The static role configured in the web service.
callingnumber	The telephone number or the directory number of the calling party. It depends on the dial plan configured on Unified CM and the transformations provisioned on the translation pattern. The number is represented by the regular expression *[+]?[0-9A-D*#]{1,48}*. It can be in E.164 number format.
callednumber	The telephone number or the directory number of called party. It depends on the dial plan configured on Unified CM and the transformations provisioned on the translation pattern. The number is represented by the regular expression *[+]?[0-9A-D*#]{1,48}*. It can be in E.164 number format.
transformedcgpn	The transformed calling number. The result of the calling number transformation applied at the translation pattern.
transformedcdpn	The transformed called number. The result of the called number transformation applied at the translation pattern.
resource-id	The name of the resource to which the dynamic roles are mapped and for which the authorization decision is to be made.
action-id	The action a subject is permitted to perform on a resource.
triggerpointtype	Specifies where the external call control is triggered. For Unified CM 8.0(1), only TranslationPattern is supported.

Example

The following is an example call request scenario:

Scenario

User A with directory number 9725550101 calls user B by dialing 50102. Cisco Unified CM is configured to allow five digit dialing for internal calls. The dialed number, 50102, is transformed into +19725550102 on the translation pattern 5XXXX.

Unified CM sends an XACML call routing request to the web service when processing the dialed digits on the translation pattern. The call routing request is as follows:

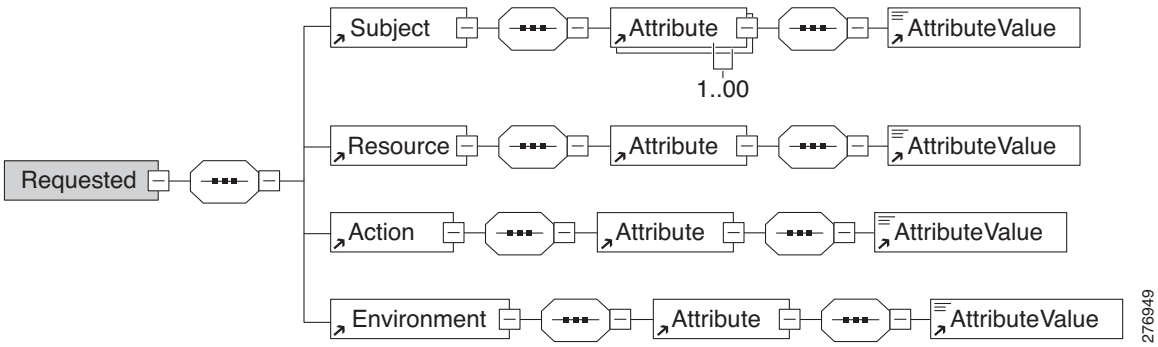
```
POST /pdp/AuthorizationEndPoint HTTP/1.1
Host: 10.89.81.55:8080
Accept: */*
Content-type: text/xml; charset=ISO-8859-1
methodName: isRoleAccessAllowed
User-Agent: CiscoUCM-HttpClient/1.0
Connection:Keep-Alive
Content-Length: 1704
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os">
<Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:role-id"
DataType="http://www.w3.org/2001/XMLSchema#string" Issuer="requestor">
<AttributeValue>CISCO:UC:UCMPolicy</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:Cisco:uc:1.0:callingnumber"
DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>+19725550101</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:Cisco:uc:1.0:callednumber"
DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>50102</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:Cisco:uc:1.0:transformedcgpn"
DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>+19725550101</AttributeValue>
</Attribute>
<Attribute AttributeId="urn:Cisco:uc:1.0:transformedcdpn"
DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>+19725550102</AttributeValue>
</Attribute>
</Subject>
<Resource>
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#anyURI">
<AttributeValue>CISCO:UC:VoiceOrVideoCall</AttributeValue>
</Attribute>
</Resource>
<Action>
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#anyURI">
<AttributeValue>any</AttributeValue>
</Attribute>
</Action>
<Environment>
<Attribute AttributeId="urn:Cisco:uc:1.0:triggerpointtype"
DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>translationpattern</AttributeValue>
</Attribute>
</Environment>
</Request>
```

XACML Request Schema

Figure 10-2 displays the schema diagram for the XACML call routing request.

Figure 10-1 XACML Schema



Schema text

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.cisco.com/ExternalCallControl"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.cisco.com/ExternalCallControl" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="Request">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Subject"/>
                <xs:element ref="Resource"/>
                <xs:element ref="Action"/>
                <xs:element ref="Environment"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Action">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Attribute"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Attribute">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="AttributeValue"/>
            </xs:sequence>
            <xs:attribute name="Issuer" type="xs:NMTOKEN" use="optional"/>
            <xs:attribute name="DataType" type="xs:string" use="required"/>
            <xs:attribute name="AttributeId" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="AttributeValue">
        <xs:complexType mixed="true"/>
    </xs:element>
    <xs:element name="Environment">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Attribute"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Resource">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Attribute"/>
```

```
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Subject">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Attribute" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Call Routing Response

This section provides information on the call routing response message. The following topics are covered:

Description

On receiving a call routing request, the Policy Decision Point evaluates the request and generates an XACML response. The XACML response is returned to Unified CM as the body of a 200 OK message. The XACML response includes one of the following policy decsions:

- Permit—The call is allowed.
- Deny—The call is denied.
- Indeterminate—Unable to evaluate the policy for the request. Routes the call as specified in the Call Treatment on Failure configuration that is set on the external call control profile.
- Not Applicable—Cannot find policy applicable to the request. Routes the call as specified in the Call Treatment on Failure configuration that is set on the external call control profile.

The XACML may contain a call instruction XML (CIXML) obligation attribute that gives additional instructions for Unified CM to route the call or apply other special treatments. Example for obligations include directives to play a particular greeting before connecting a call and diverting a call to a different extension than the one that was dialed.

The CIXML obligation must be consistent with the policy decision. If it is not, then Unified CM obeys the policy decision and not the obligation. For example, an obligation set to divert the call is ignored if the policy decision is to deny that call.

The CIXML consists of a call routing directive and one or more optional sub-elements, with or without the attributes. The call routing directives are call routing action verbs and are defined in CIXML elements. A call routing directive is a mandatory element in an obligation. An obligation can have only one call routing directive.

Format

The following is the call routing response format:

```
HTTP/1.1 <Status-Code and Reason Phase>
Server: <product | comment>
Connection:Keep-Alive
Keep-Alive: timeout = <timeou value>  max  = <count>
Location: <new-URI>
Date: <date and time>
Content-Length: <msg body leng>
```

```
<?xml encoding=”UTF-8” version=”1.0”?>
<Response>
<Result ResourceId="CISCO:UC:VoiceOrVideoCall">
<Decision>[decision value]</Decision>
<Status> <StatusCode Value="[status code]"/>
<StatusMessage>[status message]</StatusMessage>
<StatusDetail>[status details]</StatusDetail>
</Status>
<Obligations>
  [other obligations]
<Obligation FulfillOn="[fulfil on value]"
ObligationId="urn:cisco:cepm:3.3:xacml:policy-attribute">
<AttributeAssignment AttributeId="[name of attribue]">
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
<cixml version="1.0>
[routing directive[sub-element]]
</cixml>
</AttributeValue>
</AttributeAssignment>
</Obligation>
</Obligations>
</Result>
</Response>
```

For CIXML format, see CIXML Format, page 10-12.

Parameter

This section provides description of various parameters in the request format.

Parameters	Description
HTTP Post Message	
<Status-Code and Reason Phase>	One of the following: • 200 OK—HTTP response received successfully • 302 Found—Unified CM resends the request using the new request URI returned in the location header. • 307 Temporary Redirect—Unified CM follows the new location specified in the location header and re-sends the same request. Unified CM does not support cache-control and expire-header in a 307 responses. • 4xx or 5xx—Client or server errors encountered. Unified CM follows the Call Treatment on Failure configuration specified in the external call control profile to route call. **Note** For details on error handling, see Error Handling, page 10-23.
<product | comment >	The software used by the web service to handle the request.
Connection: Keep-Alive	This connection header indicates that the web service supports persistent connection and the same connection can be used by Unified CM to send the next request.

Parameters	Description
Keep-Alive: timeout = <timeout value> max = <count>	The timeout value indicates the duration the server will keep the connections live. Unified CM sends a Keep-Alive message before the timeout. The max parameter is ignored by Unified CM. If the persistent connection has not been timed-out, the Keep-Alive header may not be included in the response.
Date: <date and time>	The date and time at which the response is sent from the server.
Location: <new-URI>	The location header should be included in a 302, 303 or 307 response. Unified CM resends the request to the new location specified by new-URI.
Content-Type: text/xml; charset=<char coding>	Unified CM only accepts responses with the message body in text/xml format encoded in us-ascii or utf-8. For example: *text/xml; charset=utf-8*
Content-Length	The Content-Length field indicates the size of the entity-body, in decimal number of octets.
XACML Message	
ResourceId	The name of the resource to which the dynamic roles are mapped and for which the authorization decision is to be made.
decision-value	This is the decision of the policy evaluation. It takes one of the following values: • Permit—The requested call is allowed. • Deny—The requested call is denied. • Indeterminate—The web service is unable to evaluate policies for the requested call. Reasons for such inability include missing attributes from request, network errors while retrieving policies, syntax errors in the decision requests or in the policies, and so on. • NotApplicable—The web service cannot find a policy applicable to the request
StatusCode	The status code represents the status of the policy decision result. Unified CM recognize the following status codes: • `urn:oasis:names:tc:xacml:1.0:status:missing-attribute`—some attributes are missing in the XACML request message. • `urn:oasis:names:tc:xacml:1.0:status:ok`—the request is successfully processed. • `urn:oasis:names:tc:xacml:1.0:status:processing-error`—some processing errors are encountered when evaluating policies for the request. • `urn:oasis:names:tc:xacml:1.0:status:syntax-error`—the request contains elements with syntax errors.
StatusMessage	The status message describing the status code. For example, “Request is successful”.
StatusDetail	Additional message description for the status code.
FulfillOn	The fulfillon value is the effect for which this obligation must be fulfilled. It is either Permit or Deny.

Parameters	Description
other obligations	Unified CM ignores all obligations other than those described in this document.
name of attribute	The name of the obligation as defined in the policy.

For description of CIXML parameters, see CIXML Parameter, page 10-14.

CIXML Format

The following are the formats for the various call routing obligations (directives):

- Continue directives:
  - Simple continue—Call routes normally to the current destination:

```
<cixml version="1.0">
<continue></continue>
</cixml>
```

  - Continue with modified calling and called number:

```
<cixml version="1.0">
<continue>
<modify callingnumber=[number]callednumber=[number]/>
</continue>
</cixml>
```

  - Continue with greeting:

```
<cixml version="1.0">
<continue >
<greeting identification=[id]/>
</continue>
</cixml>
```

- Divert directives:
  - Divert to a different number or to voicemail:

```
<cixml version="1.0">
<divert>
<destination>[number]|voicemail</destination>
</divert>
</cixml>
```

  - Divert to a number with modified calling/called numbers:

```
<cixml version="1.0">
<divert >
<destination>[number]</destination>
<modify callingnumber=[number] callednumber=[number]/>
</divert>
</cixml>
```

  - Divert with call history reset:

```
<cixml version="1.0">
<divert >
<destination>[number]</destination>
```

```
<resetcallhistory> resetLastHop|resetAllHops </resetcallhistory>
</divert>
</cixml>
```

– Divert to a chaperone:

```
<cixml version="1.0">
<divert >
<destination>[number]</destination>
<reason> chaperone </reason>
</divert>
</cixml>
```

Note A chaperone is a special type of end-user who monitor calls between two parties.

- Reject directives:
  - Simple reject—Unified CM rejects the call, and the caller hears fast busy tone:

```
<cixml version="1.0">
<reject> </reject>
</cixml>
```

  - Reject with announcement:

```
<cixml version="1.0">
<reject>
<announce identification=[id] />
</reject>
</cixml>
```

  - Reject with reason string:

```
<cixml version="1.0">
<reject>
<reason>[string]</reason>
</reject>
</cixml>
```

The web service can return an XACML call routing response with policy decision of Permit or Deny without including a CIXML as an obligation. In this case, Unified CM assumes a default CIXML element that corresponds to the policy decision. The following are the default CIXML elements:

- If policy decision is permit, then:

```
<cixml version="1.0"> <continue> </continue> </cixml>
```

- If policy decision is deny, then:

```
<cixml version="1.0"> <reject> </reject> </cixml>
```

CIXML Parameter

This section provides description of various parameters in CIXML.

Parameters	Description
<continue> </continue>	The continue routing directive instructs Unified CM to proceed with the call routing process. On receiving the continue directive, Unified CM proceeds to find the best match with the resulting digits of the translation pattern. The web service can overwrite the result of the translation pattern by including a sub-element called <modify>. The continue directive has the following sub-elements: • *<greeting identification=ann-id/>*—This sub-element specifies that an announcement will be played to the calling party before routing the call to the called party. The **ann-id** should match the one provisioned in the Unified CM administration Media Resources-Announcements page. • *<modify callingNumber=number calledNumber=number/>*—This sub-element specifies that the calling number or called number is modified with the number(s) provided. The web service overwrites the calling and called party transformation that is configured for the translation pattern. Unified CM changes the calling or called number(s) to the numbers that are provided in the directive. If the number is not included in the directive, the configuration for the route pattern or translation pattern applies.
<reject></reject>	The reject routing directive instructs Unified CM to reject the call. This is the only valid call routing directive if the policy decision is “Deny”. The reject directive has the following sub-elements: • *<announce identification=ann-id/>*—this sub-element specifies that an announcement will be played to the calling party before the call is cleared. The **ann-id** should match the one provisioned in the Unified CM administration Media Resources-Announcements page. • *<reason> reason-string </reason>*—this sub-element specifies the reason for the call to be rejected. When a reason is included, Unified CM will log warning alarm. An email notification may be sent if it is configured through Unified CM RTMT tool.

Parameters	Description
<divert></divert>	The divert routing directive instructs Unified CM to redirect the call to another destination. The divert directive has the following sub-elements: • *<destination> number | voicemail</destination>*—This sub-element is mandatory for the divert directive. This sub-element specifies the destination to which the call is diverted. The destination can be a number routable in Unified CM, such as a directory number, a hunt pilot, or a route pattern. The destination can also be the string voicemail. If voicemail is specified as the destination, Unified CM will redirect the call to the voice mail box of the original called party. The call will be released if the original called party does not have a voicemail box. • *<reason>chaperone</reason>*—this sub-element specifies the reason for the call diversion. Chaperone indicates the destination is a chaperone line or a hunt list of chaperone lines. When this reason is included, the destination device will be enabled for chaperone features. For more information on chaperone feature, refer the *Cisco Unified Communications Manager Features and Services Guide, release 8.0(1)*. • *<modify callingNumber=number calledNumber=number/>*—This sub-element specifies that the calling number or called number is modified with the number(s) provided. The web service overwrites the calling and called party transformation that is configured for the translation pattern. Unified CM changes the calling or called number(s) to the numbers that are provided in the directive. If the number is not included in the directive, the configuration for the route pattern or translation pattern applies. • *<resetCallHistory> resetLastHop | resetAllHops </resetCallHistory>*—this sub-element specifies the method to reset call history. The following options are available: – *resetLastHop*: erase the last called party number from the call history. – *resetAllHops*: erase all previous called party numbers from the call history.

Example

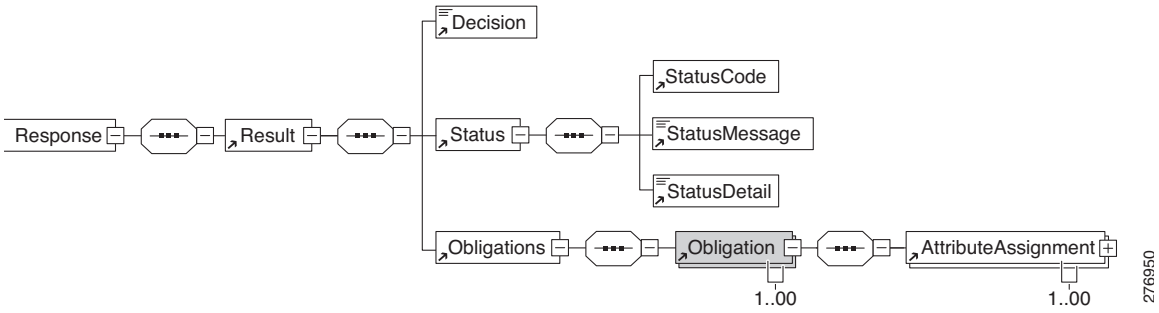
The following is an example for a call routing response.

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Connection: Keep-Alive
Keep-Alive: timeout = 1000  max = 100
Transfer-Encoding: chunked
Date: Mon, 08 Jun 2009 16:50:21 GMT
<?xml encoding="UTF-8" version="1.0"?>
<Response>
   <Result><Decision>Permit</Decision>
   <Obligations>
      <Obligation FulfillOn="Permit" obligationId="continue.simple">
         <AttributeAssignment AttributeId="Policy:continue.simple">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
             <cixml version="1.0>
               <continue></continue>
             </cixml>
         </AttributeValue>
      </AttributeAssignment>
       </Obligation>
     </Obligations>
   </Result>
</Response>
```

XACML Response Schema

Figure 10-2 displays the schema diagram for the XACML call routing response.

Figure 10-2 XACML Response Schema



Schema Text

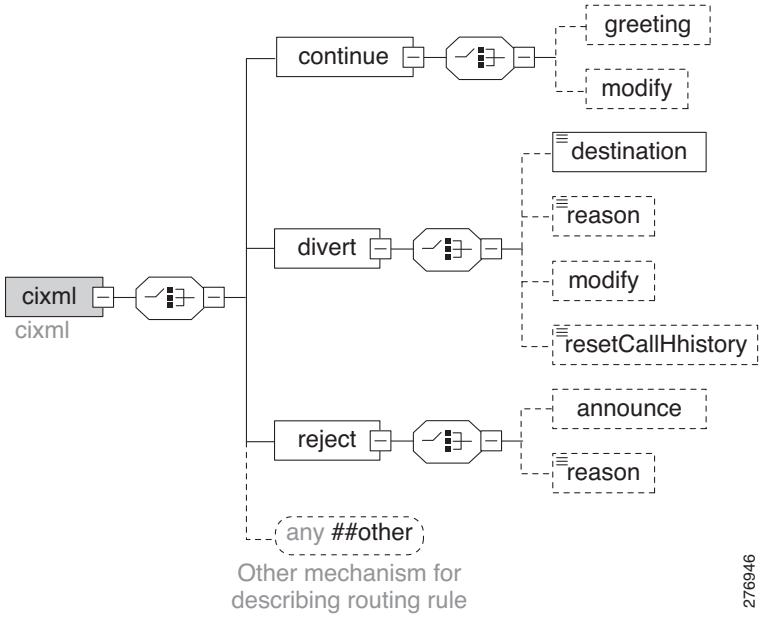
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="AttributeAssignment">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="AttributeValue"/>
            </xs:sequence>
            <xs:attribute name="AttributeId" type="xs:NMTOKEN" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="AttributeValue">
        <xs:complexType mixed="true">
```

```
            <xs:attribute name="DataType" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Decision">
        <xs:complexType mixed="true"/>
    </xs:element>
    <xs:element name="Obligation">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="AttributeAssignment" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="FulfillOn" type="xs:NMTOKEN" use="required"/>
            <xs:attribute name="ObligationId" type="xs:NMTOKEN" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Obligations">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Obligation" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Response">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Result"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Result">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Decision"/>
                <xs:element ref="Status"/>
                <xs:element ref="Obligations"/>
            </xs:sequence>
            <xs:attribute name="ResourceId" type="xs:NMTOKEN" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Status">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="StatusCode"/>
                <xs:element ref="StatusMessage"/>
                <xs:element ref="StatusDetail"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="StatusCode">
        <xs:complexType>
            <xs:attribute name="Value" type="xs:NMTOKEN" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="StatusDetail">
        <xs:complexType mixed="true"/>
    </xs:element>
    <xs:element name="StatusMessage">
        <xs:complexType mixed="true"/>
    </xs:element>
</xs:schema>
```

CIXML Schema

Figure 10-3 displays the schema diagram for the CIXML.

Figure 10-3 Call Instruction XML Schema



Schema Text

```
<xs:schema targetNamespace="http://www.cisco.com/ExternalCallControl"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.cisco.com/ExternalCallControl" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="cixml">
        <xs:annotation>
            <xs:documentation>cixml</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice>
                <xs:element name="continue">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="greeting" minOccurs="0">
                                <xs:complexType>
                                    <xs:attribute name="identification" use="required">
                                        <xs:simpleType>
                                            <xs:restriction base="xs:string"/>
                                        </xs:simpleType>
                                    </xs:attribute>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="modify" minOccurs="0">
                                <xs:complexType>
                                    <xs:attribute name="callingNumber" type="phoneNumber"
use="optional"/>
                                    <xs:attribute name="calledNumber" type="phoneNumber"
use="optional"/>
                                </xs:complexType>
                            </xs:element>
```

```
                        </xs:choice>
                    </xs:complexType>
                </xs:element>
                <xs:element name="divert">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="destination" type="destinationNumber"/>
                            <xs:element name="reason" type="reasonCode" minOccurs="0"/>
                            <xs:element name="modify" minOccurs="0">
                                <xs:complexType>
                                    <xs:attribute name="callingNumber" type="phoneNumber"
use="optional"/>
                                    <xs:attribute name="calledNumber" type="phoneNumber"
use="optional"/>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="resetCallHistory" type="resetMode"
minOccurs="0"/>
                        </xs:choice>
                    </xs:complexType>
                </xs:element>
                <xs:element name="reject">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="announce" minOccurs="0">
                                <xs:complexType>
                                    <xs:attribute name="identifiaction" type="xs:string"
use="optional"/>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="reason" type="xs:string" minOccurs="0"/>
                        </xs:choice>
                    </xs:complexType>
                </xs:element>
                <xs:any namespace="##other" processContents="lax" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>Other mechanisms for describing routing
rule</xs:documentation>
                    </xs:annotation>
                </xs:any>
            </xs:choice>
            <xs:attribute name="version" type="xs:integer"/>
            <xs:attribute name="id" type="xs:token"/>
        </xs:complexType>
    </xs:element>
    <xs:simpleType name="phoneNumber">
        <xs:annotation>
            <xs:documentation>phone number digits ( +, 0-9A-D,*,#)</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:pattern value="[+]?[0-9A-D*#]{1,48}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="reasonCode">
        <xs:annotation>
            <xs:documentation>reasonCode</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="unknown"/>
            <xs:enumeration value="user-busy"/>
            <xs:enumeration value="no-answer"/>
            <xs:enumeration value="unavailable"/>
            <xs:enumeration value="unconditional"/>
            <xs:enumeration value="time-of-day"/>
```

```
            <xs:enumeration value="time-of-day"/>
            <xs:enumeration value="do-not-disturb"/>
            <xs:enumeration value="deflection"/>
            <xs:enumeration value="follow-me"/>
            <xs:enumeration value="out-of-service"/>
            <xs:enumeration value="away"/>
            <xs:enumeration value="blocked"/>
            <xs:enumeration value="chaprone"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="resetMode">
        <xs:annotation>
            <xs:documentation>resetMode</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="resetLastHop"/>
            <xs:enumeration value="resetAllHops"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="destinationNumber">
        <xs:union>
            <xs:simpleType>
                <xs:restriction base="phoneNumber">
                    <xs:minInclusive value="1"/>
                    <xs:maxExclusive value="1"/>
                </xs:restriction>
            </xs:simpleType>
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="voiceMail"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:union>
    </xs:simpleType>
</xs:schema>
```

Keep-Alive Message

Unified CM periodically sends Keep-Alive messages to the web service. This maintains persistent connection with the web service and determines if the PDP engine of the web service is in service.

Request Message

Unified CM sends the following HTTP HEAD request to the web service as the Keep-Alive message:

```
HEAD [Request-URI]HTTP/1.1
Connection: Keep-Alive
```

where the Request-URI is the URI provisioned in the external call control profile.

Example request message:

```
HEAD /pdp/AuthorizationEndPoint HTTP/1.1
Host: cepm1.cisco.com:8080
Accept: */*
Content-type: text/xml; charset=ISO-8859-1
methodName: isRoleAccessAllowed
User-Agent: CiscoUCM-HttpClient/1.0
Connection:Keep-Alive
```

Response Message

The web service sends the following response to the Keep-Alive message:

```
HTTP/1.1 200 OK
Connection: Keep-Alive
Keep-Alive: timeout = <TO> max = <count>
```

When a Keep-Alive header is included in a response message, Unified CM sends a subsequent keep-alive message within the specified timeout range. When a Keep-Alive header is not present in the response, Unified CM assumes there is no connection timeout at the web service. Unified CM ignores the max count value.

Example 200 OK response message:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Connection: Keep-Alive
Keep-Alive: timeout = 20000  max = 100
Content-Length: 0
Date: Mon, 08 Jun 2009 16:50:12 GMT
```

Message Timer

The Unified CM administrator can set a call routing request timer. The call routing request timer is the maximum duration that Unified CM waits for a call routing or Keep-Alive response from the web service after sending the call routing or Keep-Alive request.

The call routing request timer is provisioned in the external call control profile. The timer can be set in the range of 1000 to 5000 milliseconds.

If the call routing request timer field in the external call control profile is left blank, Unified CM takes the cluster-wide value specified by the External Call Control Call Routing Request Timer service parameter. The value for the service parameter can be set to a number in the range of 1000 to 5000 milliseconds. The default value is 2000 milliseconds.

Connections

This section provides information on the various connection scenarios between Unified CM and the web service. The following topics are included:

Note HTTP Proxy: Cisco Unified CM does not support proxy for web services.

Multiple Connections

Unified CM establishes multiple connections to the web service to support multiple parallel requests. The number of connections may be increased at run time to support high call rates. The initial connection count and the maximum connection count to PDP can be configured in Unified CM via the service parameters.

For more information refer to Unified CM administration guides at:

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

Redundant Web Services

Unified CM allows two web services to be provisioned in an external call control profile. A primary web service and a secondary web service. The two web services can be provisioned to operate in the following two modes:

- Active-standby mode—This is the default mode. All call routing requests will be sent to the active web service. The primary web service is the active web service when it is in service. The secondary web service becomes active when the primary web service goes out of service.
- Load balance mode—Call routing requests will be sent to primary and secondary web service alternatively with the round-robin algorithm. The load balance mode is enabled on the external call control profile.

Persistent Connection and Keep-Alive

Unified CM maintains persistent connections with the web service by sending Keep-Alive messages to the web service before the connection timeout timer expires. Using the HTTP HEAD method the Keep-Alive messages are sent to the same URI as the call routing request message.

Unified CM sends an initial Keep-Alive message when it establishes a connection to the web service. The subsequent Keep-Alive is sent when the connection timeout timer expires. The Keep-Alive timer is reset each time a response to a Keep-Alive or a call routing request is received.

The minimum connection timeout value is 1000 milliseconds and the maximum is 20000 milliseconds.

The connection timeout value for Unified CM cannot be configured. The web service can include a Keep-Alive header in the response message to change the connection timeout value within the 1000 to 20000 milliseconds range. When a Keep-Alive header is included, Unified CM resets the Keep-Alive timer to 100 milliseconds less than the timeout value returned. If the Keep-Alive header is not present in the response, Unified CM assumes there is no connection timeout at the web server and the Keep-Alive timeout value is then set to 20000 milliseconds.

Redirection

Unified CM supports the following redirections from web services in response to call routing requests:

- 302 Found—The requested call routing policy resides temporarily under a different URI. Unified CM resends the request using the new URI returned in the location header.
- 307 Temporary Redirect—The requested call routing policy resides temporarily under a different URI. Unified CM resends the request using the new URI returned in the location header. Unified CM does not support expire-header in the 307 response.

A new connection is created if the new location is on a different server and no existing connection can be reused. The maximum number of redirections Unified CM follows for a request is two. Unified CM does not cache redirected locations.

Error Handling

This section provides information on the various error handling scenarios. The following topics are included:

Web Service Connection Failure

A web service connection error occurs when Unified CM fails to establish a connection with the web service. The following reasons may cause this failure:

- The web service is not in service.
- Unified CM fails to authenticate the web service.
- The web service fails to authenticate Unified CM.
- Slow responses from the web service that cause Unified CM to time out for two consecutive call routing or Keep-Alive requests.

Unified CM handles this failure with the following actions:

- Issues a ConnectionFailureToPDP error alarm.
- Switches to standby web service for call routing requests, if two URIs are provisioned in the external call control profile to operate in active-standby mode.
- Starts sending all call routing requests to the other web service that Unified CM still has good connections with, if two URIs are provisioned in the external call control profile to operate in load balance mode.
- Retries to establish connections to the web service.
- If no web service is available for call routing request, then follows the Call Treatment on Failure configuration as set on the external call control profile to route the call.

Unified CM Timeout Awaiting Response from Web Service

The routing request timer is the maximum time in milliseconds that Unified CM waits for the response from the web service for a call routing request. The routing request timer can be provisioned in an external call control profile in the range of 1000 to 5000 milliseconds. If the timer is not set in the external call control profile, the cluster wide service parameter “External Call Control Routing Request Timer” takes effect. The default value for the timer is 2000 milliseconds.

Unified CM takes the following actions when the routing request timer expires before receiving the call routing response:

- Issues an AwaitingResponseFromPDPTimeout error alarm.

- Routes the call following the Call Treatment on Failure configuration set on the external call control profile.

Error Response from Web Service

The web service can return a 4XX or 5XX error response to Unified CM to indicate invalid call routing requests or internal errors when processing a request from Unified CM.

Unified CM takes the following actions for a 4XX or 5XX error response:

- Issues a FailureResponseFromPDP error alarm.
- Routes the call following the Call Treatment on Failure configuration on the external call control profile.

Web Service Parse Request Error

When there are errors either in the request from the Unified CM or in processing the request, the web service states these errors in the XACML response. The following status codes may be returned:

- Missing-attribute—Some mandatory attributes not found.
- Processing-error—Errors encountered when evaluating policies for the request.
- Syntax-error—The request contains elements with syntax errors.

Unified CM takes the following actions when the response from the web service contains the status indicating request errors:

- Issues an ErrorParsingResponseFromPDP warning alarm.
- Routes the call by following the call routing directive in the response.

Unified CM Parse Response Error

This error occurs when Unified CM fails to parse the response from the web service. The following errors may cause this failure:

- XACML missing mandatory attributes or error in parsing XACML mandatory attributes
- CIXML missing mandatory attributes or error in parsing CIXML mandatory attributes
- Error in parsing the CIXML optional attributes
- XACML syntax error
- CIXML syntax error

Unified CM takes the following actions when it fails parsing the response:

- XACML missing mandatory attributes, or error in parsing XACML mandatory attributes:
  - Issues an ErrorParsingDirectiveFromPDP error alarm.
  - Routes the call following the “Call Treatment on Failure” configuration on the external call control profile.
- XACML missing mandatory attributes, or error in parsing CIXML mandatory attributes:
  - Issues an ErrorParsingDirectiveFromPDP error alarm.

  - Routes the call by following the Call Treatment on Failure configuration set on the external call control profile.
- CIXML missing optional attributes, or XACML or CIXML syntax errors
  - Issues an ErrorParsingResponseFromPDP warning alarm.
  - Routes the call by following the call routing directive in the response.

Web Service Evaluation Request Error

The web service may return the following decision in XACML in the call routing response:

- Indeterminate—The web service is unable to evaluate the policy for the request
- NotApplicable—The web service cannot find a policy applicable to the request.

Unified CM takes the following actions when one of these policy decisions is returned:

- Issues an ErrorParsingDirectiveFromPDP error alarm.
- Routes the call following the Call Treatment on Failure configuration on the external call control profile.

Security

This section provides information on the security and authentication setup between Unified CM and web service. The following topics are included:

- Authentication, page 10-25
- Certificate and Key, page 10-26
- Transport Layer Security Version, page 10-26
- Cipher Specification, page 10-26

Authentication

When a secure connection is desired, a HTTPS URI must be provisioned in the Unified CM. Unified CM does not support upgrade of an HTTP connection to HTTPS.

When HTTPS is provisioned, Unified CM uses mutual authentication with a self-signed certificate or a CA issued certificate to communicate to the web service. Unified CM conducts the following verifications when authenticating the server:

- Verification of host—Checks if the certificate subject name matches the host name of the server.
- Verification of peer—Checks if the signature of the certificate is issued by the trust CA in the trust store, or if it matches the imported certificates in the trust store for a self-signed certificate.

Certificate and Key

Generation and Exchange

If mutual authentication using self-signed certificates is required, the Unified CM administrator should generate the required certificate for the web service to import. The administrator of the web service should also generate the certificate for Unified CM to import.

Certificate Format

All certificates must be in Privacy Enhanced Mail (PEM) format or convertible to PEM format.

Transport Layer Security Version

Unified CM supports Transport Layer Security (TLS) version1 for HTTPs connections.

Cipher Specification

In mutual authentication both Unified CM and the web service send the change cipher specification message to notify the receiving party that subsequent records are protected under the just-negotiated CipherSpec and keys.

The following Cipher Spec is supported in Unified CM for external call control:

```
Cipher Spec: TLS_RSA_WITH_AES_256_CBC_SHA (0x000035)
```

INDEX

Symbols

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

W

X