AXL Serviceability API Programming

This chapter describes the implementation of AXL-Serviceability APIs . Cisco Unified CallManager Real-Time information, Performance Counters, and Database information exposure occurs through an AXL-Serviceability 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 API downloads and AXL requests and responses found in this document, refer to http://www.cisco.com/pcgi-bin/dev_support/access_level/product_support. You must have a Cisco CCO account and password to access this URL.

This chapter covers the following topics:

- Introduction
- Data Model
- AXL-Serviceability APIs Ports
- Real-Time Information (RisPort)
- Authentication
- Rate Control
- Server Query Service
- Service Interface
- Log Collection Service
- CDR on Demand Service

Introduction

By exposing Cisco Unified CallManager real-time information, performance counter, and database information, customers can write customized applications. AXL-Serviceability 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. AXL-Serviceability APIs represent one server component of the Cisco Unified CallManager Serviceability product.

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

The following enhancements exist in Serviceability for AXL in Cisco Unified CallManager Release 5.0:

- You can access AXL using Serviceability
  (Serviceability > Tools AXL Web Service).

- Serviceability provides an option to activate/deactivate AXL Web Service
  (Serviceability > Tools > Service Activation).

- Serviceability provides an option to start and stop AXL Web Service
  (Serviceability > Tools > Control Center - Feature Services).

- Serviceability provides an option to change trace levels for AXL
  (Serviceability > Trace > Configuration).

The following list gives the Serviceability services that are exposed via AXL-Serviceability APIs:

- Perfmon Data Collection

  In Cisco Unified CallManager 5.0 the Perfmon Data Collection service is exposed through a new framework called Perfmon Infrastructure and System Information Library. The APIs are the same; however, the information provided through these APIs in Cisco Unified CallManager Release 5.0 will be similar but not exactly the same. Windows performance counters are no longer available and the URL will be different from previous releases for each of the SOAP services.

- Real-Time Information (RisPort) — There are no changes in this section.

The following four services are new in Cisco Unified CallManager Release 5.0:

- Server Query Service
- Service Interface
- Log Collection Service
- CDR on Demand Service

Data Model

AXL-Serviceability APIs are based on SOAP with XML request and response messages. They must conform to the structure of a SOAP Envelope element. Although SOAP is a standard protocol, many of its data model aspects are left 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 AXL-Serviceability APIs use.

Downloading Serviceability SOAP WSDL Files

Cisco Unified CallManager Release 5.0 serviceability SOAP WSDL files can be downloaded from the Cisco Unified CallManager server directly by simply entering a URL on the web browser. In each of the following examples "servername" must be replaced by an appropriate server IP address.

PerfmonPort SOAP requests service definitions are located at:

https://servername:8443/perfmonservice/services/PerfmonPort?wsdl

RisPort SOAP requests service definitions are located at:

https://servername:8443/realtimeservice/service/RisPort?wsdl

LogCollectionService SOAP requests service definitions are located at:

https://servername:8443/logcollectionservice/services/LogCollectionPort?wsdl

DimeGetFile SOAP requests service definitions are located at:

https://servername:8443/logcollectionservice/services/DimeGetFileService?wsdl

ControlCenterServices SOAP requests service definitions are located at:

https://servername:8443/controlcenterservice/services/ControlCenterServicesPort?wsdl

SOAPMonitorService SOAP requests service definitions are located at:

https://servername:8443/realtimeservice/services/SOAPMonitorService?wsdl

Clients of AXL-Serviceability 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 AXL-Serviceability 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 replaced with an appropriate server IP address.

SOAP Binding

AXL-Serviceability API SOAP binding information is well specified in the binding section of the Serviceability SOAP WSDL files. Binding specifications apply to both request and response messages. SOAP Binding covers the aspects explained in the following sections.

Character Encoding

AXL-Serviceability 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. AXL-Serviceability APIs also sets "text/xml; charset='utf-8'" as the value of the Content-Type response header field. Internally, AXL-Serviceability APIs processes the data by using UCS-2 Unicode code page.

Binding Style

AXL-Serviceability APIs uses remote procedure call (RPC) binding style. In SOAP, the word operation refers to method or function. Each AXL-Serviceability API operation call gets modeled as an RPC 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 which 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. AXL-Serviceability APIs use the standard HTTP as its transport. Clients use the POST verb to send requests to AXL-Serviceability APIs.

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

Encoding Rule

AXL-Serviceability 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.

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

For performance reasons, AXL-Serviceability 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 AXL-Serviceability API data types schema is:

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

AXL-Serviceability APIs qualify the first body entry in the response, which represents the call-return, with this target namespace. Similarly, clients of AXL-Serviceability 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 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 is the operation name. The struct contains accessor element members that represent input parameters. Operations with no parameters have no members in the struct. Names of accessor elements are the same as the names of the input parameters. Values of 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

AXL-Serviceability 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 AXL-Serviceability 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 AXL-Serviceability 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. AXL-Serviceability 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 AXL-Serviceability 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 AXL-Serviceability API operations. Every request for the AXL-Serviceability 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. AXL-Serviceability 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, AXL-Serviceability APIs reply with a fault and a fault code value that is set to the **mustUnderstand** value.

The following example shows an AXL-Serviceability 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 is 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 are the same as 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, contain no member elements in the call-return struct.

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

*OperationName*Response

OperationName represents a placeholder that refers to the name of the operation specified in the corresponding request message. This format is specific only for the AXL-Serviceability APIs implementation.

The target namespace described in the "Encoding Rule" section on page 2-4, qualifies the call-return. AXL-Serviceability APIs return HTTP status 200 when the operation succeeds.

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

Fault Message

When an AXL-Serviceability 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

AXL-Serviceability 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. AXL-Serviceability APIs do not use the Header element. See the "SOAP Header" section on page 2-5 for details.

Client

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

Server

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

FaultString

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

FaultActor

AXL-Serviceability APIs do not set this sub-element. Note that a proxy or intermediary server must include this sub-element if it generates a fault message.

Detail

If an AXL-Serviceability API parses and processes the request body and determines that an error occurs afterwards, it includes the detailed error information in the detail sub-element.

If AXL-Serviceability APIs do not include the detail sub-element in the fault element, the error does not relate to the request body. Data types defined in the AXL-Serviceability APIs WSDL files specify the encoding rule for the detail sub-element. The following fragments of the file describe the types:

```
...
64<complexType name='CallInfoType'>
65 <sequence>
66     <element name='FileName' type='xsd:string'/>
67     <element name='LineNo' type='xsd:int'/>
68     <element name='ErrorCode' type='xsd:unsignedInt'/>
69     <element name='Function' type='xsd:string'/>
70     <element name='Params' type='xsd:string'/>
71 </sequence>
72</complexType>
73
74<complexType name='ErrorInfoType'>
75 <sequence>
76     <element name='Version' type='xsd:string'/>
77     <element name='Time' type='xsd:time'/>
78     <element name='ProcId' type='xsd:unsignedInt'/>
79     <element name='ThreadId' type='xsd:unsignedInt'/>
80     <element name='ArrayOfCallInfo' type='tns:ArrayOfCallInfoType'/>
81 /sequence>
82</complexType>
...
128<complexType name='ArrayOfCallInfoType'>
129 <complexContent>
130     <restriction base='SOAP-ENC:Array'>
131         <sequence>
132             <element name='CallInfo'
133                 type='tns:CallInfoType' minOccurs='1' maxOccurs='unbounded'/>
134         </sequence>
135     </restriction>
136 </complexContent>
137</complexType>
```

AXL-Serviceability 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 AXL-Serviceability 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 shows a successful AXL-Serviceability 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 shows a failed AXL-Serviceability 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

AXL-Serviceability APIs use the following XML namespaces:

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

  This is the namespace URI for the SOAP envelope.

- http://schemas.xmlsoap.org/soap/encoding/

  This is the namespace for the SOAP-recommended encoding rule which is based on XML Schema.

- http://schemas.cisco.com/ast/soap/

  This is the namespace URL for AXL-Serviceability API data types as defined in the WSDL file.

AXL-Serviceability APIs Ports

This section explains the interface of SOAP service built-in ports in detail.

PerfmonPort

PerfmonPort comprises several operations that allow clients to do the following perfmon-related tasks:

- Collect perfmon counter data.

  AXL-Serviceability 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.

PerfmonListCounter

This operation returns the list of Perfmon objects and counters in a particular host.

Request Format

The PerfmonListCounter operation takes a single parameter:

- **Host**
  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 is an example of a PerfmonListCounter 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:PerfmonListCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <Host xsi:type="xsd:string">nozomi</Host>
```

```
            </q1:PerfmonListCounter>
        </soap:Body>
</soap:Envelope>
```

Response Format

PerfmonListCounter returns information that describes the hierarchical structure of Perfmon objects and counters. The body entry includes an ArrayOfObjectInfo element. The following fragments from the AXL-Serviceability APIs WSDL file describe the types that this response uses:

```
...
106  <complexType name='ArrayOfObjectInfoType'>
107      <complexContent>
106          <restriction base='SOAP-ENC:Array'>
109              <sequence>
110                  <element name='ObjectInfo'
111                      type='tns:ObjectInfoType' minOccurs='1' maxOccurs='unbounded'/>
112              </sequence>
113          </restriction>
114      </complexContent>
115  </complexType>
```

The ArrayOfObjectInfo element comprises an array of ObjectInfo elements that have the following type:

```
...
56  <complexType name='ObjectInfoType'>
57      <sequence>
58          <element name='Name' type='tns:ObjectNameType'/>
59          <element name='MultiInstance' type='xsd:boolean'/>
60          <element name='ArrayOfCounter' type='tns:ArrayOfCounterType'/>
61      </sequence>
62  </complexType>


...
24  <simpleType name='ObjectNameType'>
25      <restriction base='string'/>
26  </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:

```
...
44  <complexType name='CounterType'>
45      <sequence>
46          <element name='Name' type='tns:CounterNameType'/>
47      </sequence>
48  </complexType>
32  <simpleType name='CounterNameType'>
33      <restriction base='string'/>
34  </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>
```

PerfmonListInstance

This 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.

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:PerfmonListInstance xmlns:q1="http://schemas.cisco.com/ast/soap/">
                    <Host xsi:type="xsd:string">nozomi</Host>
                    <Object xsi:type="xsd:string">Process</Object>
                </q1:PerfmonListInstance>
        </soap:Body>
</soap:Envelope>
```

Response Format

PerfmonListInstance returns an element named ArrayOfInstance. The type for this element specifies ArrayOfInstanceType, which is an array of Instance elements. The following fragments from AXL-Serviceability APIs .WSDL file explain the types that this response uses:

```
...
84<complexType name='ArrayOfInstanceType'>
85  <complexContent>
86      <restriction base='SOAP-ENC:Array'>
87          <sequence>
88              <element name='Instance'
89                  type='tns:InstanceType' minOccurs='0' maxOccurs='unbounded'/>
90          </sequence>
91      </restriction>
92  </complexContent>
93</complexType>
```

> **Note** ArrayOfInstanceType can have 0 (zero) Instance elements, in which case the requested object is not of a multi-instance object.

```
...
50<complexType name='InstanceType'>
51  <sequence>
52      <element name='Name' type='tns:InstanceNameType'/>
53  </sequence>
54</complexType>
```

The type for Instance element specifies InstanceType. It represents a structure with a single-element member: Name.

```
...
28<simpleType name='InstanceNameType'>
29  <restriction base='string'/>
30</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:

Example

```
<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:PerfmonListInstanceResponse xmlns:m="http://schemas.cisco.com/ast/soap/">
            <ArrayOfInstance SOAP-ENC:arrayType="m:InstanceType[]">
                <Instance>
                    <Name>Idle</Name>
                </Instance>
                <Instance>
                    <Name>System</Name>
```

```
                            </Instance>
                            <Instance>
                                <Name>SMSS</Name>
                            </Instance>
                            <Instance>
                                <Name>CSRSS</Name>
                            </Instance>
                            <Instance>
                                <Name>WINLOGON</Name>
                            </Instance>
                            <Instance>
                                <Name>SERVICES</Name>
                            </Instance>
                            <Instance>
                                <Name>_Total</Name>
                            </Instance>
                            ...
                        </ArrayOfInstance>
                    </m:PerfmonListInstanceResponse>
                </SOAP-ENV:Body>
            </SOAP-ENV:Envelope>
```

PerfmonQueryCounterDescription

This operation returns the help text of a particular counter.

Request Format

PerfmonQueryCounterDescription takes the following parameter:

- **Counter**—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

PerfmonQueryCounterDescription returns an element 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.

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>
```

PerfmonOpenSession

Client programs submit this operation to get a session handle from the AXL-Serviceability APIs. The client needs a session handle to do the session-based perfmon counter data collection. The session handle is a universally unique identifier that is used once, which guarantees that no duplicate handles exist. AXL-Serviceability APIs keep the opened handles in cache. If no activity occurs on a handle for 25 hours, the AXL-Serviceability API removes the handle and renders it invalid.

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 AXL-Serviceability APIs can remove the handle from cache. Clients can dynamically add new counters to the session handle and remove counters from it 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" ?>
<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 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" 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>
```

PerfmonAddCounter

This operation adds an array of counters to a session handle.

Request Format

PerfmonAddCounter takes the following parameters:

- **SessionHandle**
  The type is SessionHandleType which is derived from xsd:string. It contains the session handle previously opened by the previous PerfmonOpenSession operation.

- **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 fragments from AXL-Serviceability APIs describe the types that this request uses:

```
...
95<complexType name='ArrayOfCounterType'>
96  <complexContent>
97      <restriction base='SOAP-ENC:Array'>
98          <sequence>
99              <element name='Counter'
100                 type='tns:CounterType' minOccurs='1' maxOccurs='unbounded'/>
101         </sequence>
102     </restriction>
103 </complexContent>
104</complexType>
```

✎
Note   ArrayOfCounterType expects at least one Counter element in the array.

```
...
44<complexType name='CounterType'>
45  <sequence>
46      <element name='Name' type='tns:CounterNameType'/>
47  </sequence>
48</complexType>
```

CounterType represents a structure, and it has a single element member: Name.

```
...
32<simpleType name='CounterNameType'>
33  <restriction base='string'/>
34</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 multi-reference accessors.

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:PerfmonAddCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <SessionHandle xsi:type="xsd:string">
                {1A490F1E-D82C-403F-9CF0-C4D4ABD6FF3E}
            </SessionHandle>
            <ArrayOfCounter href="#id1"/>
        </q1:PerfmonAddCounter>
        <soapenc:Array id="id1"
            xmlns:q2="http://schemas.cisco.com/ast/soap/"
            soapenc:arrayType="q2:CounterType[3]">
            <Item href="#id2" />
            <Item href="#id3" />
            <Item href="#id4" />
        </soapenc:Array>
        <q3:CounterType id="id2" xsi:type="q3:CounterType"
            xmlns:q3="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Segments/sec</Name>
        </q3:CounterType>
        <q4:CounterType id="id3" xsi:type="q4:CounterType"
            xmlns:q4="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Connections Reset</Name>
        </q4:CounterType>
        <q5:CounterType id="id4" xsi:type="q5:CounterType"
            xmlns:q5="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Connections Active</Name>
        </q5:CounterType>
    </soap:Body>
</soap:Envelope>
```

Response Format

The following example shows that the PerfmonAddCounter returns no output:

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/"
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <m:PerfmonAddCounterResponse xmlns:m="http://schemas.cisco.com/ast/soap/"/>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

If PefmonAddCounter fails to add one or more counters that are specified in the request, AXL-Serviceability 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, AXL-Serviceability 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.

PerfmonRemoveCounter

This operation removes an array of counters from a session handle.

Request Format

PerfmonRemoveCounter takes the following parameters:

- **SessionHandle**
  The type is SessionHandleType which is derived from xsd:string. It contains the session handle opened previously by the PerfmonOpenSession operation.

- **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 is an example of PerfmonRemoveCounter request with three counters in the ArrayOfCounter parameter. This example uses single-reference accessor style.

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: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 multi reference accessors.

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:PerfmonRemoveCounter xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <SessionHandle xsi:type="xsd:string">
                {1A490F1E-D82C-403F-9CF0-C4D4ABD6FF3E}
            </SessionHandle>
            <ArrayOfCounter href="#id1" />
        </q1:PerfmonRemoveCounter>
        <soapenc:Array id="id1" xmlns:q2="http://schemas.cisco.com/ast/soap/"
            soapenc:arrayType="q2:CounterType[3]">
            <Item href="#id2" />
            <Item href="#id3" />
            <Item href="#id4" />
        </soapenc:Array>
        <q3:CounterType id="id2" xsi:type="q3:CounterType"
            xmlns:q3="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Segments/sec</Name>
        </q3:CounterType>
        <q4:CounterType id="id3" xsi:type="q4:CounterType"
            xmlns:q4="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Connections Reset</Name>
        </q4:CounterType>
        <q5:CounterType id="id4" xsi:type="q5:CounterType"
            xmlns:q5="http://schemas.cisco.com/ast/soap/">
            <Name xsi:type="xsd:string">\\nozomi\tcp\Connections Active</Name>
        </q5:CounterType>
    </soap:Body>
</soap:Envelope>
```

Response Format

PerfmonRemoveCounter returns no data in the response as shown by the following example:

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:PerfmonRemoveCounterResponse xmlns:m="http://schemas.cisco.com/ast/soap/"/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

If the PefmonRemoveCounter operation fails to remove one or more counters that the request specifies, the AXL-Serviceability API replies with a fault response with semantics similar to PerfmonAddCounter. If some of the counters specified in the request get removed successfully, while others failed to be removed, the AXL-Serviceability 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.

PerfmonCollectSessionData

This operation collects the perfmon data for all counters that have been added to the query handle.

Request Format

PerfmonCollectSessionData takes the following parameter:

- **SessionHandle**
  The type is SessionHandleType which is derived from xsd:string. It contains the session handle opened previously by the PerfmonOpenSession operation.

The following example shows a PerfmonCollectSessionData 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:PerfmonCollectSessionData xmlns:q1="http://schemas.cisco.com/ast/soap/">
         <SessionHandle xsi:type="xsd:string">
            {FB343D6D-AA6E-4EDB-9CFA-63A5A3ED6405}
         </SessionHandle>
      </q1:PerfmonCollectSessionData>
   </soap:Body>
</soap: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 AXL-Serviceability APIs .WSDL show the types that this response uses:

```
...
117<complexType name='ArrayOfCounterInfoType'>
118  <complexContent>
119      <restriction base='SOAP-ENC:Array'>
120          <sequence>
121              <element name='CounterInfo'
122                  type='tns:CounterInfoType' minOccurs='1' maxOccurs='unbounded'/>
123          </sequence>
124      </restriction>
125  </complexContent>
126</complexType>
```

ArrayOfCounterInfoType has one or more CounterInfo elements in the array. The CounterInfo element includes the following type:

```
...
36<complexType name='CounterInfoType'>
37  <sequence>
```

```
38      <element name='Name' type='tns:CounterNameType'/>
39      <element name='Value' type='xsd:long'/>
40      <element name='CStatus' type='xsd:unsignedInt'/>
41 </sequence>
42</complexType>
...
32<simpleType name='CounterNameType'>
33 <restriction base='string'/>
34</simpleType>
```

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 PerfmonCollectSessionData response:

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:PerfmonCollectSessionDataResponse
xmlns:m="http://schemas.cisco.com/ast/soap/">
        <ArrayOfCounterInfo SOAP-ENC:arrayType="m:CounterInfoType[]">
            <CounterInfo>
                <Name>\\nozomi\tcp\Segments/sec</Name>
                <Value>0</Value>
                <CStatus>0</CStatus>
            </CounterInfo>
            <CounterInfo>
                <Name>\\nozomi\tcp\Connections Reset</Name>
                <Value>6</Value>
                <CStatus>0</CStatus>
            </CounterInfo>
            <CounterInfo>
                <Name>\\nozomi\tcp\Connections Active</Name>
                <Value>23</Value>
                <CStatus>0</CStatus>
            </CounterInfo>
        </ArrayOfCounterInfo>
    </m:PerfmonCollectSessionDataResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

PerfmonCloseSession

This operation closes the session handle that the PerfmonOpenSession previously retrieved.

Request Format

PerfmonCloseSession takes the following parameter:

- **SessionHandle**
  The type is SessionHandleType which is derived from xsd:string. It contains the session handle
  previously opened by the PerfmonOpenSession operation.

The following example shows a PerfmonCloseSession 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:PerfmonCloseSession xmlns:q1="http://schemas.cisco.com/ast/soap/">
<SessionHandle xsi:type="xsd:string">
{01944B7E-183F-44C5-977A-F31E3AE59C4C}
</SessionHandle>
        </q1:PerfmonCloseSession>
    </soap:Body>
</soap:Envelope>
```

Response Format

The following example shows that the PerfmonCloseSession does not return data in the response:

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:PerfmonCloseSessionResponse xmlns:m="http://schemas.cisco.com/ast/soap/"/>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

PerfmonCollectCounterData

This 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

PerfmonCollectCounterData 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" ?>
<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:PerfmonCollectCounterData xmlns:q1="http://schemas.cisco.com/ast/soap/">
            <Host xsi:type="xsd:string">nozomi</Host>
            <Object xsi:type="xsd:string">Processor</Object>
        </q1:PerfmonCollectCounterData>
    </soap:Body>
</soap:Envelope>
```

Response Format

PerfmonCollectCounterData 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" 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:PerfmonCollectCounterDataResponse
            xmlns:m="http://schemas.cisco.com/ast/soap/">
            <ArrayOfCounterInfo SOAP-ENC:arrayType="m:CounterInfoType[]">
                <CounterInfo>
                    <Name>\\nozomi\Processor(0)\% Processor Time</Name>
                    <Value>99</Value>
                    <CStatus>0</CStatus>
                </CounterInfo>
                <CounterInfo>
                    <Name>\\nozomi\Processor(0)\% User Time</Name>
                    <Value>0</Value>
                    <CStatus>0</CStatus>
                </CounterInfo>
                <CounterInfo>
                    <Name>\\nozomi\Processor(0)\% Privileged Time</Name>
                    <Value>0</Value>
                    <CStatus>0</CStatus>
                </CounterInfo>
                <CounterInfo>
                    <Name>\\nozomi\Processor(0)\Interrupts/sec</Name>
```

```
                    <Value>525</Value>
                    <CStatus>0</CStatus>
                </CounterInfo>
                ...
            </ArrayOfCounterInfo>
        </m:PerfmonCollectCounterDataResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Real-Time Information (RisPort)

Selecting Cisco Unified CallManager Real-Time Information

Request Format

SOAP Action and Envelope Information
This operation allows clients to perform Cisco Unified CallManager device-related queries. HTTP header should have following SOAP action for these queries.

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

Query information includes an Envelope as follows:

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
3.  xmlns:tns="http://schemas.cisco.com/ast/soap/"
4.  xmlns:types="http://schemas.cisco.com/ast/soap/encodedTypes"
5.  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6.  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.

```
7.  <soap:Header>
8.  <tns:AstHeader id="id1">
9.  <SessionId xsi:type="xsd:string">SessionId</SessionId>
10. </tns:AstHeader>
11. </soap:Header>
```

Selection Criteria
Selection criteria type, which is a Cisco Unified CallManager Selection criteria, follows the SOAP header.

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

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

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

Maximum Devices Specification
This example specifies how many maximum devices can be returned for search criteria.

```
17. <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'.

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

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

```
19. <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.'

```
20. <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.

```
21. <NodeName xsi:type="xsd:string" />
```

Specify Selection type whether it is IP Address/Name

```
22. <SelectBy xsi:type="tns:CmSelectBy">Name</SelectBy>
```

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.

```
23. <SelectItems href="#id2" />Name or IP</tns:CmSelectionCriteria
24. <soapenc:Array id="id2" soapenc:arrayType="tns:SelectItem[2]">
25. <Item href="#id3"/><Item xsi:null="1"/>
26. </soapenc:Array>
27. <tns:SelectItem id="id3" xsi:type="tns:SelectItem">
28. <Item xsi:type="xsd:string"/></tns:SelectItem>
29. </soap:Body>
30. </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='unbounded'/>
       </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='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>
```

Selecting CTI Real-Time Information

Request Format

SOAP Action

This operation allows client to perform a CTI manager-related query.

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:

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

```
6.  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

Session ID

The following example is a SOAP header that contains a session ID. The client sets a unique session ID.

```
7.  <soap:Header>
8.  <tns:AstHeader id="id1">
9.  <SessionId xsi:type="xsd:string">jSessionId</SessionId>
10. </tns:AstHeader>
11. </soap:Header>
12.
13. <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
14. <tns:SelectCtiItem><StateInfo xsi:type="xsd:string" /><CtiSelectionCriteria
href="#id1" /></tns:SelectCtiItem>
15. <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:

```
16. <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:

```
17. <CtiMgrClass xsi:type="tns:CtiMgrClass">Line</CtiMgrClass>
```

Status of CTI Item Search

The following example specifies the Status of class on which to search:

```
18. <Status xsi:type="tns:CtiStatus">Any</Status>
```

Server Name for Search

The following example specifies the server name on which the search is performed:

```
19. <NodeName xsi:type="xsd:string" />
```

Type of Search

The following example specifies the type of selection:

```
20. <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:

```
21. <AppItems href="#id2" />Name /IP</tns:CtiSelectionCriteria>
22. <soapenc:Array id="id2" soapenc:arrayType="tns:SelectAppItem[2]">
23. <Item href="#id3" /><Item xsi:null="1" /></soapenc:Array>
24. <tns:SelectAppItem id="id3" xsi:type="tns:SelectAppItem">
25. <AppItem xsi:type="xsd:string"/>
26. </tns:SelectAppItem>
27. </soap:Body>
28. </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>
```

How to Get All Device Information in a Large System

The Serviceability-SOAP interface has StateInfo as part of the response. This is the unique string that tells the client the state of the Device server information. Clients have to give this string from the first request to the next request for the same selection criteria. In that case, clients will get the response with the "NoChange" element set to true as part of CmNode. This means that the server information state has not changed from the previous state, which means no new registration or deregistration of devices has happened. This saves lot of time for clients and servers since the server is telling the client that there is no change from the previous state. If "NoChange" in the response is set to false, this indicates the server information has changed. In this case, the client needs to get the real-time information from the server and update the client information on the devices.

```
<!-- 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>
```

As part of the response a maximum of 200 devices is provided per response, as in the wsdl.

```
<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 is done to limit the response buffer to the first 200 devices per server even though the client may have given search criteria to get all the devices in a large deployment. Searches in call processing servers are expensive in terms of CPU cycles. Call processing servers need to service IP phone calls. The device requests from clients should consume less memory and CPU resources. Device requests cannot exceed 20% of CPU resources in call processing servers. Many clients may be requesting this information, but we have limited the information to the first 200 devices.

To get all configured device information, clients must use the AXL-DB API. Use the device information from the AXL-DB API in "SelectItems" of the "CmSelectionCriteria" Serviceability SOAP APIs to get first 200 device's real-time information.

```
<complexType name=" ">
<sequence><element name="MaxReturnedDevices" type="xsd:unsignedInt"/><element
name="Class" type="tns:DeviceClass"/><element name="Model"
type="xsd:unsignedInt"/><element name="Status" type="tns:CmDevRegStat"/><element
name="NodeName" type="xsd:string"/><element name="SelectBy"
type="tns:CmSelectBy"/><element name="SelectItems" type="tns:SelectItems"/>
</sequence>
</complexType>

<complexType name="SelectItems">
<complexContent>
<restriction base="SOAP-ENC:Array"><sequence><element name="SelectItem"
type="tns:SelectItem" minOccurs="0" maxOccurs="200"/></sequence>
</restriction>
</complexContent>
</complexType>
```

The most efficient way to get the list of devices is to use "Executesqlqueryreq()" of the AXL-DB API. One could use an SQL equivalent to "Select name from device where tkclass = 1" to get all phones, then iterate through the list sending 200 at a time to the AXIS-SOAP API.

Device requests per minute can not exceed 15 per minute (by default) to the Cisco Unified CallManager server. So space client requests so that they do not exceed 15 requests per minute. If the client requests exceed 15 per minute, the server will respond with a SOAP fault. Based on this fault, a SOAP client can adjust the request rate. These rates are expected to take less than 20% of the CPU resources, which will not affect the Cisco Unified CallManager's ability to respond to IP phone call requests.

Selecting SIP Device Real Time Information

Request Format

SOAP Action

This operation allows clients to perform Cisco Unified CallManager SIP device related queries. HTTP header has 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 optional State Info is followed by selection criteria type CmSelectionCriteriaSIP.

```
<CmSelectionCriteriaSIP href="#id0"/>
</ns1:SelectCmDeviceSIP>
```

CmSelectionCriteriaSIP is composed of the following:

- **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 is 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. No name specified will search in all servers in cluster.

  ```
  <NodeName xsi:type="xsd:string" xsi:nil="true"/>
  ```

- **SelectBy**—specifies the selection type whether it is IP Address/Name.

  ```
  <SelectBy xsi:type="xsd:string">Name</SelectBy>
  ```

- **SelectItems**—the array of items for which search criteria is specified. Following specifies array that contains IP Address or Device Name of the items for which we need the real time status.

```
<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 returned by the SOAP server. Each node has a sequence of search information 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 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 information defined below.

```
<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 is an example of 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 get defined as part of the AXL-DB WSDL file. Send your queries to API question mailer on these interfaces.

Authentication

The following sections describe the currently used authentication.

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 is a clear-text authentication that is not secure. The Basic provides an advantage because it is widely implemented by many platforms, and most HTTP client agents support it. Basic authentication can be secure if the encryption, such as SSL, is used.

Secure

When you install/upgrade Cisco Unified CallManager, the HTTPS self-signed certificate, httpscert.cer, automatically installs on the default website that hosts the Cisco Unified CallManager 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 Cisco Unified CallManager applications support HTTPS, which ensures the identity of the server:

 • Cisco Unified CallManager Administration
 • Cisco Unified CallManager Serviceability
 • Cisco Unified IP Phone User Option Pages
 • Cisco Unified CallManager Bulk Administration
 • Cisco Unified CallManager Auto-Register Phone Tool
 • Cisco Unified CallManager CDR Analysis and Reporting
 • Cisco Unified CallManager Trace Collection Tool

- Cisco Unified CallManager Real Time Monitoring Tool.

For more information, refer to the *Cisco Unified CallManager Security Guide, Release* 5.0.

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," the access to SOAP APIs gets denied.

Rate Control

The AXL-Serviceability Interface includes a request rate control mechanism that monitors the request rates for the complete system. If the request rate is more than supported, a "SOAP Fault" is sent to the client and the request is blocked. This rate control is enabled for "Real-time Data requests" and "Perfmon Data Requests." These rates are controlled through "Enterprise parameters" as shown in Figure 2-1. The system provides Help for these parameter configurations as shown in Figure 2-2.

Figure 2-1   Rate Control Enterprise Parameters
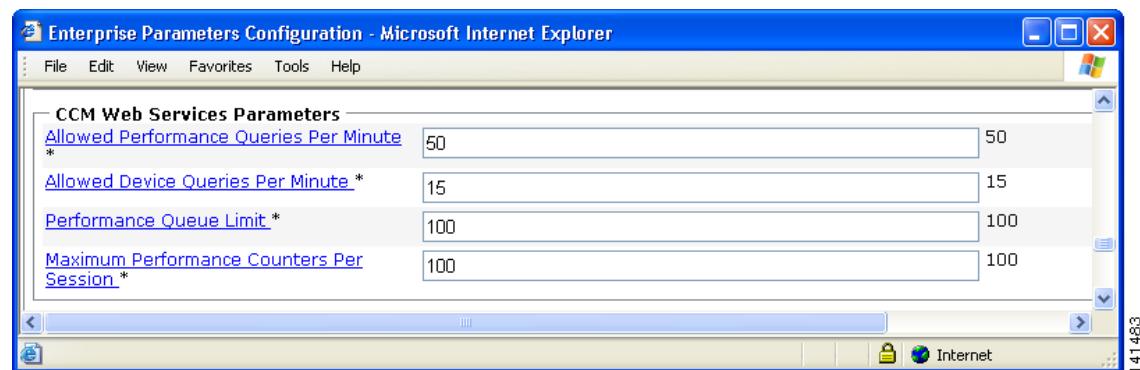
Figure 2-2   Help for Rate Control Parameters
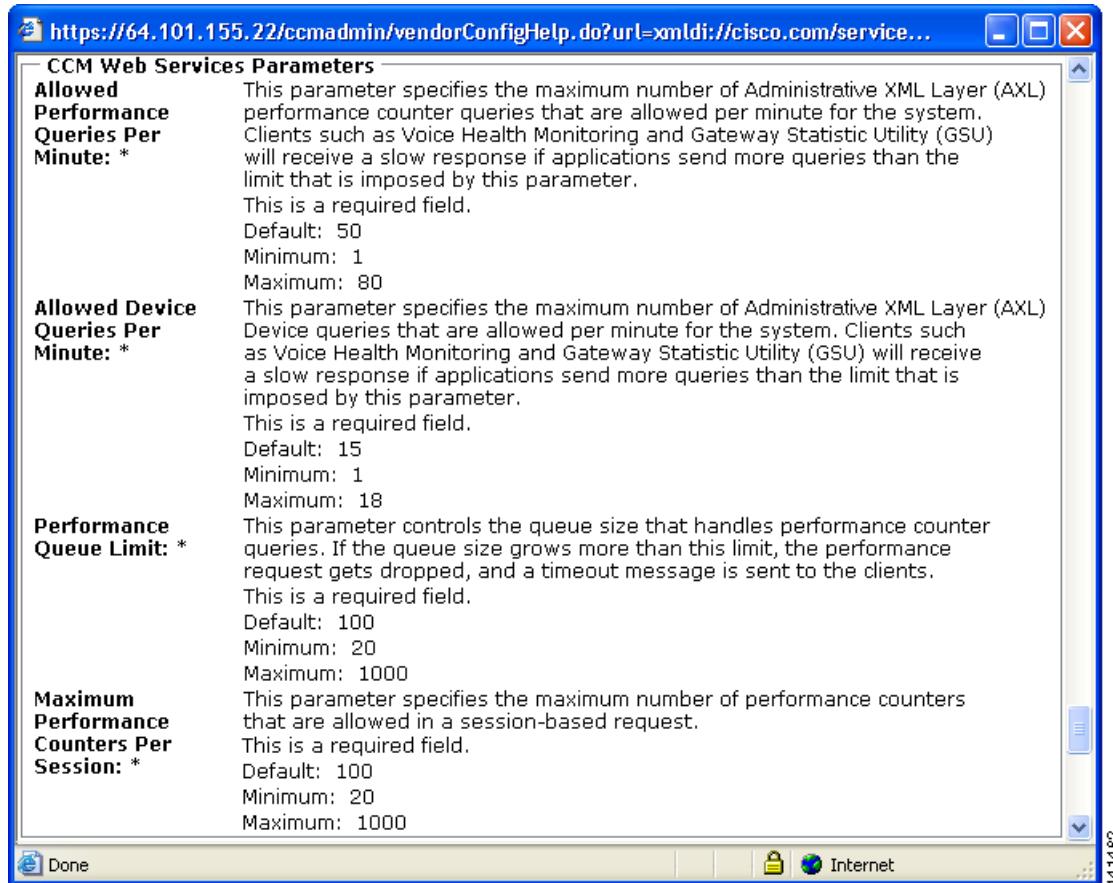
Server Query Service

The following information is exported 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 =5.0.1.0-212

The getServerInfo operation consists of 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

In the request an ArrayOfHosts definition is specified and the response provides the server information for the list of hostnames specified in the array.

```
- <wsdl:message name="getServerInfoRequest">
  <wsdl:part name="Hosts" type="impl:ArrayOfHosts" />
  </wsdl:message>
```

Response Format

The response consists of 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" />
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
<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" />
</sequence>
</complexType>
```

Faults

The Server will send a fault for "Error message context is NULL." This error will 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 xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>172.19.240.90</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 xsi:type="soapenc:Array" soapenc:arrayType="ns1:ServerInformation[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
     <item>
     <HostName xsi:type="xsd:string">MCS-SD4</HostName>
     <os-name xsi:type="xsd:string">Linux</os-name>
     <os-version xsi:type="xsd:string">2.4.21-15.ELsmp</os-version>
     <os-arch xsi:type="xsd:string">i386</os-arch>
     <java-runtime-version xsi:type="xsd:string">1.4.2_05-b04</java-runtime-version>
     <java-vm-vendor xsi:type="xsd:string">Sun Microsystems Inc.</java-vm-vendor>
     < Active_Versions xsi:type="xsd:string"> list of rpm separated by :
     </Active_Versions>
     <In_Active_Versions xsi:type="xsd:string">list of rpm separated by :
     </In_Active_Versions>
     </item>
    </ServerInfo>
   </ns1:GetServerInfoResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Service Interface

The Service Interface allows you to do Service Deploy, Service UnDeploy, Get Service List, and perform service start and stop.

Get Static Service List

This **soapGetStaticServiceList** API allows clients to perform a query for all services static specification in the system. 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#soapGetSer
viceList"/>
    <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 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.

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: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 request example for getting static service specification list 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:GetStaticServiceList
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://schemas.cisco.com/ast/soap/">
      <ServiceInformationResponse
xsi:type="xsd:string">test</ServiceInformationResponse>
    </ns1:GetStaticServiceList>
  </soapenv:Body>
</soapenv:Envelope>
```

Response Example

The response example for getting static service specification list 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: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[50]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
          <item>
            <ServiceName xsi:type="xsd:string">Cisco Unified CallManager</ServiceName>
            <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
            <Deployable xsi:type="xsd:boolean">true</Deployable>
            <GroupName xsi:type="xsd:string">CM Services</GroupName>
            <DependentServices xsi:type="soapenc:Array"
soapenc:arrayType="xsd:string[1]">
              <item>Cisco RIS Data Collector</item>
            </DependentServices>
          </item>
          <item>
            <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>
            <DependentServices xsi:type="soapenc:Array"
soapenc:arrayType="xsd:string[1]">
              <item>Cisco RIS Data Collector</item>
            </DependentServices>
          </item>
          <item>
            <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>
            <DependentServices xsi:type="soapenc:Array"
soapenc:arrayType="xsd:string[1]">
              <item>Cisco RIS Data Collector</item>
            </DependentServices>
          </item>
          <item>
            <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>
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
                    <GroupName xsi:type="xsd:string">CM Services</GroupName>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[1]">
                      <item>Cisco RIS Data Collector</item>
                    </DependentServices>
                  </item>
                  <item>
                    <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>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[2]">
                      <item>Cisco RIS Data Collector</item>
                      <item>Cisco Unified CallManager</item>
                    </DependentServices>
                  </item>
                  <item>
                    <ServiceName xsi:type="xsd:string">Cisco Unified 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>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[1]">
                      <item>Cisco RIS Data Collector</item>
                    </DependentServices>
                  </item>
                  <item>
                    <ServiceName xsi:type="xsd:string">Cisco RIS Data Collector</ServiceName>
                    <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                    <Deployable xsi:type="xsd:boolean">true</Deployable>
                    <GroupName xsi:type="xsd:string">CM Services</GroupName>
                    <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                  </item>
                  <item>
                    <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>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[1]">
                      <item>Cisco RIS Data Collector</item>
                    </DependentServices>
                  </item>
                  <item>
                    <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>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[1]">
                      <item>Cisco RIS Data Collector</item>
                    </DependentServices>
                  </item>
                  <item>
                    <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">CM Services</GroupName>
                    <DependentServices xsi:type="soapenc:Array"
                          soapenc:arrayType="xsd:string[2]">
                      <item>Cisco RIS Data Collector</item>
                      <item>Cisco AMC Service</item>
                    </DependentServices>
                  </item>
                  <item>
                    <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>
      <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
    </item>
    <item>
      <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">CM Services</GroupName>
      <DependentServices xsi:type="soapenc:Array"
            soapenc:arrayType="xsd:string[1]">
        <item>Cisco RIS Data Collector</item>
      </DependentServices>
    </item>
    <item>
      <ServiceName xsi:type="xsd:string">Cisco CDR Repository Manager
    </ServiceName>
      <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
      <Deployable xsi:type="xsd:boolean">true</Deployable>
      <GroupName xsi:type="xsd:string">CDR Services</GroupName>
      <DependentServices xsi:type="soapenc:Array"
            soapenc:arrayType="xsd:string[1]">
        <item>Cisco RIS Data Collector</item>
      </DependentServices>
    </item>
    <item>
      <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>
      <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
    </item>
    <item>
      <ServiceName xsi:type="xsd:string">Cisco CDR Agent</ServiceName>
      <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
      <Deployable xsi:type="xsd:boolean">true</Deployable>
      <GroupName xsi:type="xsd:string">CDR Services</GroupName>
      <DependentServices xsi:type="soapenc:Array"
            soapenc:arrayType="xsd:string[1]">
        <item>Cisco RIS Data Collector</item>
      </DependentServices>
    </item>
    <item>
      <ServiceName xsi:type="xsd:string">Cisco SOAP - CDRonDemand Service
    </ServiceName>
      <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
      <Deployable xsi:type="xsd:boolean">true</Deployable>
      <GroupName xsi:type="xsd:string">CDR Services</GroupName>
      <DependentServices xsi:type="soapenc:Array"
            soapenc:arrayType="xsd:string[2]">
        <item>Cisco RIS Data Collector</item>
        <item>Cisco CDR Repository Manager</item>
      </DependentServices>
    </item>
    <item>
      <ServiceName xsi:type="xsd:string">Cisco CAR Scheduler</ServiceName>
      <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
      <Deployable xsi:type="xsd:boolean">true</Deployable>
      <GroupName xsi:type="xsd:string">CDR Services</GroupName>
      <DependentServices xsi:type="soapenc:Array"
            soapenc:arrayType="xsd:string[3]">
        <item>Cisco RIS Data Collector</item>
        <item>Cisco CDR Repository Manager</item>
        <item>Cisco CAR Web Service</item>
      </DependentServices>
    </item>
    <item>
      <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>
                <DependentServices xsi:type="soapenc:Array"
                        soapenc:arrayType="xsd:string[3]">
                  <item>Cisco RIS Data Collector</item>
                  <item>Cisco CDR Repository Manager</item>
                  <item>Cisco CAR Scheduler</item>
                </DependentServices>
              </item>
              <item>
                <ServiceName xsi:type="xsd:string">Cisco AMC Service</ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                <Deployable xsi:type="xsd:boolean">true</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="soapenc:Array"
                        soapenc:arrayType="xsd:string[1]">
                  <item>Cisco RIS Data Collector</item>
                </DependentServices>
              </item>
              <item>
                <ServiceName xsi:type="xsd:string">Cisco Unified CM SNMP Service<
                    /ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                <Deployable xsi:type="xsd:boolean">true</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
              </item>
              <item>
                <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">CM Services</GroupName>
                <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
              </item>
              <item>
                <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>
                <ServiceName xsi:type="xsd:string">Cisco DRF Master</ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                <Deployable xsi:type="xsd:boolean">true</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
              </item>
              <item>
                <ServiceName xsi:type="xsd:string">Cisco DRF Local</ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                <Deployable xsi:type="xsd:boolean">true</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
              </item>
              <item>
                <ServiceName xsi:type="xsd:string">Cisco Unified IP Phone Services<
                    /ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
                <Deployable xsi:type="xsd:boolean">true</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
              </item>
              <item>
                <ServiceName xsi:type="xsd:string">Cisco Unified CallManager</ServiceName>
                <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                <Deployable xsi:type="xsd:boolean">false</Deployable>
                <GroupName xsi:type="xsd:string">CM Services</GroupName>
                <DependentServices xsi:type="soapenc:Array"
                        soapenc:arrayType="xsd:string[1]">
```

```
          <item>Cisco RIS Data Collector</item>
        </DependentServices>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco Tftp</ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CM Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[1]">
          <item>Cisco RIS Data Collector</item>
        </DependentServices>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco Messaging Interface</ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CM Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[1]">
          <item>Cisco RIS Data Collector</item>
        </DependentServices>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco IP Voice Media Streaming App
      </ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CM Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[1]">
          <item>Cisco RIS Data Collector</item>
        </DependentServices>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco CTIManager</ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CM Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[2]">
          <item>Cisco RIS Data Collector</item>
          <item>Cisco Unified CallManager</item>
        </DependentServices>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco Unified CallManager Attendant
            Console Server</ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CTI Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[1]">
          <item>Cisco RIS Data Collector</item>
        </DependentServices>
      </item>
      <item>
        <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">CM Services</GroupName>
        <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
      </item>
      <item>
        <ServiceName xsi:type="xsd:string">Cisco Extension Mobility</ServiceName>
        <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
        <Deployable xsi:type="xsd:boolean">false</Deployable>
        <GroupName xsi:type="xsd:string">CM Services</GroupName>
        <DependentServices xsi:type="soapenc:Array"
                soapenc:arrayType="xsd:string[1]">
```

```
                        <item>Cisco RIS Data Collector</item>
                          </DependentServices>
                      </item>
                      <item>
                        <ServiceName xsi:type="xsd:string">Cisco Extended Functions</ServiceName>
                        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                        <Deployable xsi:type="xsd:boolean">false</Deployable>
                        <GroupName xsi:type="xsd:string">Voice Quality Reporter Services
                        </GroupName>
                        <DependentServices xsi:type="soapenc:Array"
                              soapenc:arrayType="xsd:string[1]">
                          <item>Cisco RIS Data Collector</item>
                          </DependentServices>
                      </item>
                      <item>
                        <ServiceName xsi:type="xsd:string">Cisco Serviceability Reporter
                        </ServiceName>
                        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                        <Deployable xsi:type="xsd:boolean">false</Deployable>
                        <GroupName xsi:type="xsd:string">CM Services</GroupName>
                        <DependentServices xsi:type="soapenc:Array"
                            soapenc:arrayType="xsd:string[2]">
                          <item>Cisco RIS Data Collector</item>
                          <item>Cisco AMC Service</item>
                          </DependentServices>
                      </item>
                      <item>
                        <ServiceName xsi:type="xsd:string">Cisco CTL Provider</ServiceName>
                        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                        <Deployable xsi:type="xsd:boolean">false</Deployable>
                        <GroupName xsi:type="xsd:string">Security Services</GroupName>
                        <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                      </item>
                      <item>
                        <ServiceName xsi:type="xsd:string">Cisco WebDialer Web Service
                        </ServiceName>
                        <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
                        <Deployable xsi:type="xsd:boolean">false</Deployable>
                        <GroupName xsi:type="xsd:string">CM Services</GroupName>
                        <DependentServices xsi:type="soapenc:Array"
                              soapenc:arrayType="xsd:string[1]">
                          <item>Cisco RIS Data Collector</item>
                          </DependentServices>
                      </item>
                      <item>
                        <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>
                        <DependentServices xsi:type="soapenc:Array"
                              soapenc:arrayType="xsd:string[1]">
                          <item>Cisco RIS Data Collector</item>
                          </DependentServices>
                      </item>
                      <item>
                        <ServiceName xsi:type="xsd:string">Cisco Certificate Authority Proxy
                            Function</ServiceName>
                        <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                        <Deployable xsi:type="xsd:boolean">false</Deployable>
                        <GroupName xsi:type="xsd:string">Security Services</GroupName>
                        <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                      </item>
                      <item>
                        <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>
                        <DependentServices xsi:type="soapenc:Array"
                              soapenc:arrayType="xsd:string[1]">
                          <item>Cisco RIS Data Collector</item>
```

```
                            </DependentServices>
                        </item>
                        <item>
                          <ServiceName xsi:type="xsd:string">Cisco SOAP - CDRonDemand Service
                           </ServiceName>
                          <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
                          <Deployable xsi:type="xsd:boolean">false</Deployable>
                          <GroupName xsi:type="xsd:string">CDR Services</GroupName>
                          <DependentServices xsi:type="soapenc:Array"
                                soapenc:arrayType="xsd:string[2]">
                            <item>Cisco RIS Data Collector</item>
                            <item>Cisco CDR Repository Manager</item>
                          </DependentServices>
                        </item>
                        <item>
                          <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>
                          <DependentServices xsi:type="soapenc:Array"
                                soapenc:arrayType="xsd:string[3]">
                            <item>Cisco RIS Data Collector</item>
                            <item>Cisco CDR Repository Manager</item>
                            <item>Cisco CAR Web Service</item>
                          </DependentServices>
                        </item>
                        <item>
                          <ServiceName xsi:type="xsd:string">Cisco CAR Web Service</ServiceName>
                          <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
                          <Deployable xsi:type="xsd:boolean">false</Deployable>
                          <GroupName xsi:type="xsd:string">CDR Services</GroupName>
                          <DependentServices xsi:type="soapenc:Array"
                                soapenc:arrayType="xsd:string[3]">
                            <item>Cisco RIS Data Collector</item>
                            <item>Cisco CDR Repository Manager</item>
                            <item>Cisco CAR Scheduler</item>
                          </DependentServices>
                        </item>
                        <item>
                          <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">CM Services</GroupName>
                          <DependentServices xsi:type="soapenc:Array"
                                soapenc:arrayType="xsd:string[1]">
                            <item>Cisco RIS Data Collector</item>
                          </DependentServices>
                        </item>
                        <item>
                          <ServiceName xsi:type="xsd:string">Cisco Unified CM SNMP Service<
                               /ServiceName>
                          <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                          <Deployable xsi:type="xsd:boolean">false</Deployable>
                          <GroupName xsi:type="xsd:string">CM Services</GroupName>
                          <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                        </item>
                        <item>
                          <ServiceName xsi:type="xsd:string">Cisco DirSync</ServiceName>
                          <ServiceType xsi:type="ns2:ServiceTypes">Service</ServiceType>
                          <Deployable xsi:type="xsd:boolean">false</Deployable>
                          <GroupName xsi:type="xsd:string">CM Services</GroupName>
                          <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                        </item>
                        <item>
                          <ServiceName xsi:type="xsd:string">Cisco AXL Web Service</ServiceName>
                          <ServiceType xsi:type="ns2:ServiceTypes">Servlet</ServiceType>
                          <Deployable xsi:type="xsd:boolean">false</Deployable>
                          <GroupName xsi:type="xsd:string">Database and Admin Services</GroupName>
                          <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                        </item>
                        <item>
```

```
                    <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">CM Services</GroupName>
                    <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                  </item>
                  <item>
                    <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">CM Services</GroupName>
                    <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                  </item>
                  <item>
                    <ServiceName xsi:type="xsd:string">Cisco Unified 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>
                    <DependentServices xsi:type="xsd:string" xsi:nil="true"/>
                  </item>
                </Services>
              </StaticServiceList>
            </ns1:GetStaticServiceListResponse>
          </soapenv:Body>
        </soapenv:Envelope>
```

Get Service Status API

This **soapGetServiceStatus** API allows clients to perform a list of deployable/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

HTTP header should have following SOAP action and envelop information:

```
<operation name="soapGetServiceStatus">
    <soap:operation
soapAction="http://schemas.cisco.com/ast/soap/action/#ControlCenterServices#soapGetSer
viceStatus"/>
    <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 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 will be 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 are the details about ServiceStatus, ReasonCod,e and ReasonCodeString.

- ServiceStatus is either Started or Stopped. If Started, StartTime gives the time 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 the service is Stopped:

  - If a deployable service is activated and stopped by admin, then its status is Stopped, the ReasonCode is -1019, and the corresponding ReasonCodeString is "Component is not running".

  - If a deployable service is deactivated, then its status is Stopped, the ReasonCode is -1068, and the corresponding ReasonCodeString is "Service Not Activated".

- If a non-deployable is stopped by admin, then its status could be Stopped with ReasonCode -1019 and the corresponding ReasonCodeString "Component is not running".

The complete listing of ReasonCode and ReasonCodeString is as 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 non-existent service name, then ReasonCode -1010 and ReasonCodeString "No such service" will appear in the response. The following is the request and response example 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

The request example for getting "Cisco Tftp" service status 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: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

The response example for getting "Cisco Tftp" service status 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: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

The request example for getting service status when providing an empty array of service names 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: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

The response for the above 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">Cisco Unified CallManager</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">Cisco Unified CallManager 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 CallManager 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">Cisco Unified CallManager 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">Cisco Unified CallManager 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">Cisco Unified CallManager 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>
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
                     <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>
```

Do Service Deployment API

This **soapDoServiceDeployment** API 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. This
API can only be used 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.

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#soapDoServ
iceDeployment"/>
    <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 2-51.

Fault

Invalid Service: Non-existing Service

If the request to activate or deactivate a service is for an invalid service name, such as a non-existent service name, then ReasonCode -1010 and ReasonCodeString "No such service" will appear in the response.

The following is the request and response example 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: Non-deployable Service

If the request to activate or deactivate a service is for a non-deployable service; a service with the Deployable attribute set tofalse in the response for getting the static service specification, such as service "SNMP Master Agent," then ReasonCode -1045 and ReasonCodeString "Operation not supported" will appear in the response.

The following is the request and response example 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, then ReasonCode -1045 and ReasonCodeString "Operation not supported" will appear in the response.

The following is the request and response example 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 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: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 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: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>
```

Control Services API

This **soapDoControlServices** API 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 doesn't allow clients to stop the following non-stop services:

- A Red Hat DB
- Cisco Tomcat
- Cisco Database Layer Monitor
- Cisco Unified CallManager Serviceability
- Cisco SOAP -Real-Time Service APIs
- Cisco SOAP -Performace Monitoring APIs
- Cisco SOAP -Log Collection APIs

The API also doesn't allow clients to provide an empty list of services when trying to start/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 .

Fault

Invalid Service: Non-existing Service

If the request of start/stop service for an invalid service name, such as a non-existing service name, then ReasonCode -1010 and ReasonCodeString "No such service" will be responded. The following is the request and response example 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: Non-stop Service

If the request of stop service for a non-stop service, such as service "Cisco Tomcat", then ReasonCode -1045 and ReasonCodeString "Operation not supported" will be responded. The following is the request and response example 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 with an empty list of services, then ReasonCode -1045 and ReasonCodeString "Operation not supported" will be responded. The following is the request and response example 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 for stop a service, and at the meantime the service status is Stopping, then ReasonCode -1045 and ReasonCodeString "Operation not supported" will be responsed. The request and response example is very similar to the example above.

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 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: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>
```

Log Collection Service

This section describes the API calls for the log collection service.

ListNodeServiceLogs

listNodeServiceLogs returns array of NodeServiceLogList. This array has the node names in the cluster and the list of service names / system log names.

Request Format

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

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>Cisco Unified CallManager 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>Cisco Unified CallManager</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 there is no node in the database, it will return a remote exception, "Query selectAllProcessNodes failed.

If there is no service list/syslog list retuned from the prefs, it will return a remote exception, No service found"/"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>
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
              <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>Cisco Unified CallManager 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>Cisco Unified CallManager</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>
```

SelectLogFiles

selectLogFiles takes FileSelectionCriteria object as an input parameter and returns FileSelectionResult.

In FileSelectionCriteria, you must specify the files you need to download. These are the parameters:

- LogCollectionService URL - http://hostname/logcollectionservice/services/LogCollectionPort
- Array of Service Names / System Log Names for which the files need to be collected
- Username -CMAdministrator
- Password - ciscocisco
- Frequency - OnDemand
- File Collection Time range - Possible values are Day, Week, Hour, Minute, Month
- File Collection Time Value - Possible values are 1 -100.
- Time Zone - Example - "Client:(GMT-8:0)Pacific Standard Time"
- Time to wait after sending a request - in milliseconds
- Local folder where the files are to copied - Example d:\soaptest
- Hostname of the server

Request Format

```xml
<?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>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>Cisco Unified CallManager 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>Cisco Unified CallManager</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>
     <item>Event Viewer-Application Log</item>
     <item>Install Logs</item>
     <item>Event Viewer-System Log</item>
     <item>Security Logs</item>
     <item>Cisco Tomcat</item>
```

Cisco Unified CallManager Developers Guide for Release 5.0

```
        </ServiceLogs>
        <SystemLogs xsi:type="xsd:string" xsi:nil="true"/>
        <SearchStr xsi:type="xsd:string" xsi:nil="true"/>
        <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="#id2"/>
        <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="id2" soapenc:root="0"
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xsi:type="ns3:RelText"
  xmlns:ns3="http://cisco.com/ccm/serviceability/soap/LogCollection/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">Days</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>
     </soapenv:Body>
    </soapenv:Envelope>
```

Response Format

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 xsi:type="soapenc:Array" soapenc:arrayType="ns2:ServiceLogs[1]"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
       <item>
        <name xsi:type="xsd:string" xsi:nil="true"/>
        <SetOfFiles xsi:type="soapenc:Array" soapenc:arrayType="ns2:file[747]">
         <item>
          <name xsi:type="xsd:string">messages</name>
          <absolutepath
xsi:type="xsd:string">/var/log/active/syslog/messages</absolutepath>
          <filesize xsi:type="xsd:string">1048783</filesize>
         </item>
         <item>
          <name xsi:type="xsd:string">messages.1</name>
          <absolutepath
xsi:type="xsd:string">/var/log/active/syslog/messages.1</absolutepath>
          <filesize xsi:type="xsd:string">1208798</filesize>
         </item>
```

```
        <item>
         <name xsi:type="xsd:string">rtmt00025.log</name>
         <absolutepath
xsi:type="xsd:string">/var/log/active/tomcat/logs/rtmt/log4j/rtmt00025.log</absolutepath>
            <filesize xsi:type="xsd:string">1000084</filesize>
            </item>
            <item>
         <name xsi:type="xsd:string">rtmt00026.log</name>
         <absolutepath
xsi:type="xsd:string">/var/log/active/tomcat/logs/rtmt/log4j/rtmt00026.log</absolutepath>
            <filesize xsi:type="xsd:string">1000104</filesize>
            </item>
         </SetOfFiles>
        </item>
      </ServiceList>
     </Node>
   </FileSelectionResult>
   <ScheduleList xsi:type="soapenc:Array" soapenc:arrayType="ns3:Schedule[0]"
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 frequency specified 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 there is matching file found, it throws a remote exception. "The File Vector from the server is null"

Example

Following is the example for FileCollection from all the services in last 5 days.

```
https://172.19.240.90:8443/logcollectionservice/services/LogCollectionPort
CMAdministrator ciscocisco OnDemand Days 5 "Client:(GMT-8:0)Pacific Standard Time"
6000000 "/var/log/SoapTest" 172.19.240.90
```

For the absolute time, the request would look like this

```
https://172.19.240.90:8443/logcollectionservice/services/LogCollectionPort
CMAdministrator ciscocisco OnDemand "11/14/04  2:15 PM" "11/15/04  2:15 PM"
"Client:(GMT-8:0)Pacific Standard Time" 6000000 "/var/log/SoapTest" 172.19.240.90
```

GetOneFile

getOneFile takes as an input parameter the absolute file name which you want to collect from the server.

The return value is the file name, but the file name will have an AXIS specific name. After the file is downloaded, you must repalce 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>
```

CDR on Demand Service

The CDR On-Demand Service is a public SOAP/HTTPS interface which is exposed to third party billing applications or customers to allow them to query the Cisco Unified CallManager 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, CDR records were stored in the CDR database and third party applications could query the database directly for the CDR records. In this release, CDRs are no longer 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 which are returned via a (s)FTP session.

The billing application can acquire a list of CDR files matching a specified time interval (get_file_list), with the maximum time span being one hour. If the application needs to retrieve CDR files spanning an interval over one hour, then multiple get_file_list requests must be made to the servlet.

Once 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 Cisco Unified CallManager configuration and other criteria. If for some reason the billing servers do not receive the CDR files, or wish to have them sent again, they can do so using the SOAP/HTTPS CDR On-Demand APIs. Once CDR files are deleted, they cannot be retrieved.

The CDR On-Demand Service provides the following features:

- API to get a list of files matching a specified time interval (get_file_list)
- API to request a specific file matching a specified filename (get_file)
- Limit of 1300 file names returned from the get_file_list API
- Specified time range cannot span over one 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 the SOAP APIs must be activated from the Service Activation page 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 Click on 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 is dependent on the CDR Repository Manager, so both must be activated.

Step 5 Click Update and wait until the page refreshes.

Tip Remember that the On-Demand Service will not function during the Maintenance period.

Security

Standard FTP or SFTP can be used to deliver the CDR files. Refer to RFC959 and RFC2228 for further details of these applications.

The CDR On-Demand Service will create either a standard FTP or SFTP session with the billing server each time a CDR file is to be sent. Exceptions are thrown whenever an error occurs on the Servlet side. In addition, all errors will be written into log files.

On the billing application side, it is recommended that billing applications implement code to catch these exceptions and display the exception string for detailed error conditions.

The two APIs that comprise the CDR On-Demand Service are described in the following sections.

get_file_list

The get_file_list API allows the application to query the CDR Repository Node for a list of all the files matching a specified time interval. The time interval of the request cannot exceed one hour. If you want a list of files spanning more than the one hour time interval allowed, then you must make multiple requests to the Servlet to acquire multiple lists of filenames.

The get_file_list API returns an array of strings containing the list of all the filenames matching the specified time interval. If no filenames exist matching the time range, then the value returned from the API call is simply null. If any time errors are encountered, exceptions are thrown. In addition, logs will be kept detailing the errors. These log files are located in the /var/log/active/tomcat/logs/soap/log4j directory.

There is also a limit of 1300 file names that 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, then you should make additional requests with the start time of the subsequent requests as the time of the last file name 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. There is no default value.

End Time

Mandatory parameter that specifies the ending time for the search interval. The format is a string: YYYYMMDDHHMM. There is no default value.

Note   The time span between Start Time and End Time must be a valid interval, but not longer than one 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.

get_file

The get_file API allows customers to request for a specific CDR file matching the specified filename. The resulting CDR file is then sent to the customer via standard ftp or secure ftp, depending on third party billing application's preference. The only constraint is 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 are thrown. In addition, logs will be kept detailing the errors. Log files are located 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 which 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 which 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 which 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 which the CDR Servlet is to send the CDR files.

File Name

Mandatory parameter (string) that specifies the filename of the CDR file 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 (s)FTP to deliver the CDR files. This is dependent on the third party billing application configuration and preferences.

Fault

The CDR On-Demand Service, throws exceptions when certain error conditions are met:

- Servlet is used during the maintenance period.
- The values entered for starting and ending time are not of the correct length –
  12 bytes in the format YYYYMMDDHHMM is the correct length.
- The starting and ending time spans an interval of more than one hour.
- The starting time is greater than or equal to the ending time (invalid interval).

- • There are no files in the CDR Repository.

- • Failed to establish (s)FTP connection to the remote node.

- • (s)FTP failed to send the files requested by the third party billing application.

The error condition will be described in the exception string that can be printed out by the billing application with toString() function.

Example

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://schemas.cisco.com/ast/soap/"
xmlns="http://schemas.xmlsoap.org/wsdl/" 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: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="ArrayOf_xsd_string">
- <complexContent>
- <restriction base="soapenc:Array">
  <attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]" />
  </restriction>
  </complexContent>
  </complexType>
  </schema>
  </wsdl:types>
  <wsdl:message name="get_fileResponse" />
- <wsdl:message name="get_file_listResponse">
  <wsdl:part name="get_file_listReturn" type="impl:ArrayOf_xsd_string" />
  </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_filelist" />
- <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="" />
- <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="http://irv3-ccm1:8080/CDRonDemandService/services/CDRonDemand" />
    </wsdl:port>
    </wsdl:service>
    </wsdl:definitions>
```