



CHAPTER 1

API Message Format

This chapter explains the format of the IP Solution Center (ISC) API error messages (1000-1099, 1100-1199, 2000-2099, and 2100-2299) specified in [Chapter 2, “System Error Messages.”](#)

The API in general is a request/response system. The input messages are error processed and any errors are reported back to the API client as part of the response. In the API, the errors are always formatted into a standard encoding scheme. The encoding scheme is basically a set of three attributes as shown in the following schema:

```
<xs:element name="error">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="code"/>
      <xs:element ref="description"/>
      <xs:element ref="detail"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

The error is used to define a fundamental error that prevented a method or operation from executing normally. The three attributes are as follows:

- “code” attribute contains a numerical status code indicating the nature of the error. The valid status codes are defined in [Chapter 2, “System Error Messages.”](#)
- “description” attribute provides a human-readable description of the error.
- “detail” attribute when populated provides additional clarifications. Many times an error could involve various components of ISC. In these situations the code and description could be the API error and the details could have information regarding an error in another component.

Important to error message processing is the context (create, delete, and so on) in which the error was found and the class in which the error was realized is found. For this reason, the API returns the operation and classname also in the response. Also shown in the following example is a block called <errors>. This is used because there can be multiple <error> for one response. This occurs very infrequently but can happen.

The important areas of the following message are highlighted in **bold**:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns0="http://www.cisco.com/cim-cx/2.0" xmlns:ns1="urn:CIM">
  <soapenv:Header>
    <ns0:message id="87855" sessiontoken="040833748184101892B5C1130544B053"
timestamp="2003-10-29T17:30:23.199Z" />
  </soapenv:Header>
  <soapenv:Body>
    <ns1:createInstanceResponse>
      <returns xsi:type="ns1:CIMReturnList" soapenc:arrayType="ns1:CIMReturn[]">
        <objectPath xsi:type="ns1:CIMObjectPath">
          <className xsi:type="xsd:string">VPNServicesModule</className>
          <errors xsi:type="ns1:CIMErrorList" soapenc:arrayType="ns1:CIMError[]">
            <error xsi:type="ns1:CIMError">
              <code xsi:type="xsd:int">1104</code>
              <description xsi:type="xsd:string">Unable to find object (Device) with value (CatIOS).
Referenced object does not exist.</description>
              <detail xsi:type="xsd:string">For input string: "CatIOS"</detail>
            </error>
          </errors>
        </objectPath>
      </returns>
    </ns1:createInstanceResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

In this example, the createInstanceResponse indicates the error was associated with a create operation. The class the create was being performed on was VPNServiceModule. The error code was 1104, which is listed numerically in [Chapter 2, “System Error Messages.”](#) The following message and the way this is specified in [Chapter 2, “System Error Messages”](#) is a concatenation of code and description. Hence, its **1104: Unable to find object (Device) with value (CatIOS). Referenced object does not exist.**

The following is an additional sample API error message from a performBatchOperation that has an action createInstance within it. This error is a Repository error as indicated by the error code 22. The description shows an error within the MPLS SR and the detail shows that it was an ORACLE deadlock.

```

<actionName xsi:type="xsd:string">createInstanceResponse</actionName>
  <objectPath xsi:type="ns1:CIMObjectPath">
    <className xsi:type="xsd:string">ServiceRequest</className>
    <errors xsi:type="ns1:CIMErrorList" soapenc:arrayType="ns1:CIMError[]">

```

```

    <error xsi:type="ns1:CIMError">
      <detail xsi:type="xsd:string">ORA-00060: deadlock detected while waiting for resource
    </detail>
      <description xsi:type="xsd:string">22 : SQL Exception while updating
    com.cisco.vpnsc.repository.mpls.RepMplsSR</description>
      <code xsi:type="xsd:int">22</code>
    </error>

```

The API also has error logs which are normally for debug purposes. The following is a sample output from an NBI log in the tmp directory of the installation. It shows the call to `getErrorMsgByName` with argument 2029. The 2029 is the error code.

The ASA Error (from SYBASE) would be shown in the detail field. Also shown is the stack trace that would be of use to the Cisco Technical Assistance Center (TAC) engineers.

FINE: `getErrorMsgByName(2029)`

Oct 29, 2003 12:15:57 PM com.cisco.vpnsc.repository.common.RepVpnscLogger severe

SEVERE: [NbiException.processException:

com.sybase.jdbc2.jdbc.SybSQLException: **ASA Error -196: Index 'MGMT_ADDR_CR' for table 'CISCO_ROUTER' would not be unique**

at com.sybase.jdbc2.tds.Tds.processEed(Tds.java:2538)

at com.sybase.jdbc2.tds.Tds.nextResult(Tds.java:1922)

at com.sybase.jdbc2.jdbc.ResultGetter.nextResult(ResultGetter.java:69)

at com.sybase.jdbc2.jdbc.SybStatement.nextResult(SybStatement.java:201)

at com.sybase.jdbc2.jdbc.SybStatement.nextResult(SybStatement.java:182)

