# TS-006
# Topology

| | |
|---|---|
| **Abstract:** | This document defines a standard for expressing network topologies. |
| **Authors:** | Haj Elfadil, Marcus Hines, Todd Law, Eric Miller, Bill Schongar, Ben Schurman |
| **Copyright:** | © 2014, Network Test Automation Forum. All rights reserved. |
| **Status:** | Release |
| **Revision:** | 1 |
| **Revision date** | August 2014 |
| **Submission:** | ntaf TS-006 |

# Legal Notices

This Specification has been created by the Network Test Automation Forum (NTAF). NTAF reserves the rights to at any time add to, amend, modify or withdraw all or any portion of this Specification.

Note: All parties who in any way intend to use this Specification for any purpose, are hereby put on notice that the possibility exists that practicing under this Specification may require the use of inventions covered by the patent rights held by third parties. By publication of this Specification NTAF makes no representation or warranty whatsoever, whether expressed or implied, that practicing under this will not infringe any third party rights, nor does NTAF make any representation or warranty whatsoever, whether expressed or implied, with respect to (1) any claim that has been or may be asserted by any third party, (2) the validity of any patent rights related to any such claim, (3) or the extent to which a license to use any such rights may or may not be available on reasonable and nondiscriminatory terms, or on any terms at all.

## Revision History

| Version | Date | By | Changes |
|---|---|---|---|
| 01 | 2014-09-05 | | First release |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

## Table of Contents



## Works Cited

Network Test Automation Forum. (2012, August). *TS-004: Inventory.* Retrieved from Ntaforum: http://ntaforum.org/resources/request.html

JSON to XML Converter. (n.d.) http://www.freeformatter.com/json-to-xml-converter.html

XML to JSON Converter. (n.d.) http://www.freeformatter.com/xml-to-json-converter.html

## 1. Background and Motivation

Topologies are fundamental to the network test industry and to the networking industry in general. Yet no standard exists for expressing topologies. The lack of a standard has led to diverse topology representations, which makes sharing of test cases very challenging. There can even be problems sharing topologies internally within large organizations.

This document proposes a standard for expressing topologies, such that topologies can be effectively shared, used, and understood, across the network test industry. The proposal focuses on the most fundamental aspects of topologies, but also allows for extensions.

## 2. Purpose

The purpose of this document is to provide a standard for the expression of topologies, specifically in the context of reporting, to give readers of reports a basic understanding of the topology used in a test. This document does not attempt to provide a comprehensive standard for all variations in network topologies, and leaves that task to other standardization efforts, either inside or outside of NTAF.

## 3. Terminology

To describe topologies, we first define the terms *node*, *interface* and *connection*.

### Node

A node is simply a point in a network. Commonly it can be a device, which may be real, simulated or virtual. Examples of nodes include real routers, switches, servers, traffic generators. Nodes may include simulated or virtual versions of these items. Nodes can also represent entire networks, subnets, and clouds, including the internet.

### Interface

An interface is a point on a node which can connect the node to the outside world. An interface may be a physical port on a router or traffic generator. An interface may also be something more abstract, such as a TCP port.

### Connections

An connection is a connection between two interfaces. Examples of connections include cables or fibers for topologies of physical connections. Connections may also represent connections which are abstract in nature, such as between two virtual machines.

## 4. JSON and XML Representations

The general conventions used to translate between JSON and XML for topology representations are given here.

1. The highest level object in JSON is an unnamed object. It contains one topology object. The highest level XML element name is the topology object.
2. It is common to define attributes for objects (for example see TS-004). In JSON attributes are name-value pairs. Attributes in XML are elements where name is the element name. Objects are expressed as sub-elements. Simple values (string, integer) are expressed as the text node of the element.

3. Items that allow multiple values, like object collections or attributes with multiple values, are expressed in JSON as arrays. When such items have only one value that value must be an array with a single element. The name of the array is the plural form of the item type. For example, an array of node objects is named “nodes”.
4. To convert JSON arrays to XML, use an XML element with same name as the JSON array. Array elements are expressed as sub-elements using the singular of the item name. For example, an array of nodes will be expressed by a <nodes> element which contains one or more <node> elements.

Examples of the above rules are given in the Use Cases section. Use case representations were first developed in JSON and then converted using this JSON to XML Converter. (There is also a corresponding XML to JSON Converter.)

# 5. Object Model

## Objects

### Topology

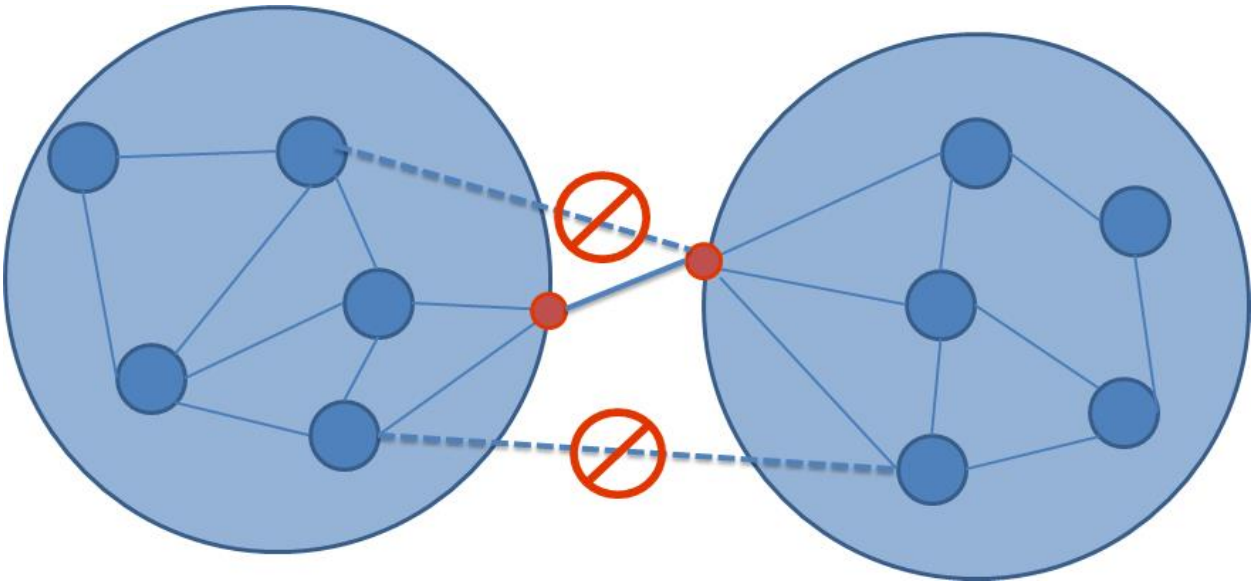
The top level container of node and connection objects. Topology is a single object. In XML the root element is <topology>. .

### Node

As mentioned above, a node is a point in a network. Nodes can contain nodes and interfaces. Nodes are collected in a "nodes" object. This nesting of nodes allows for expression of network elements at multiple levels of complexity.

Nodes can only be connected to nodes of the same nesting level or to their immediate parent. This means nested nodes can only connect to external nodes through interfaces on their parent. The allowed connections and a couple of disallowed connections are shown in the diagram below. Shown are two parent nodes, each of which contains child nodes. Child nodes can connect to each other and to their parents, plus the parent nodes can connect since they are in the same context. However, two children cannot connect; neither can one child connect to another parent. Instead, the child connects to a parent interface which can then connect to the outside world. See the Nested Node use case in Use Cases section for an example of how this works.



### Interface

Interfaces are contained within nodes. Nodes are not strictly required to have interfaces. However, nodes without interfaces cannot be connected, so any node that contributes to the geometry of a topology will have at least one interface. Likewise, not every interface is required to be associated with an connection, but the only interfaces of interest to a topology will be connected via connections. Interfaces are collected in an "interfaces" object.

Interfaces may also be nested within other interfaces. Such nesting allows for expressing subinterfaces.

One or more interfaces can be used as management ports. Interfaces that are management ports are marked with a management port attribute.

### Connection

As stated above, a connection represents a connection between interfaces. Connections may be modeled in two ways. First, connections may be represented by setting the destination attribute on the connected interfaces. It is expected that this mechanism is sufficient for most cases. Connections may also be represented by connection objects. Using connection objects is useful for complex cases where connections carry attributes or must be bundled together. Connections are collected in a "connections" object.

Connections may be nested inside other connections which allows for expression of complex connection situations.

## Attributes

All objects can have a name attribute. The name attribute is mandatory for all objects except topology. The name also serves as the object identifier and so must be unique within its context. Nodes and connection names must be unique within a topology. Interface names must be unique within a node.

All objects also support an alias attribute. Like the name attribute the alias attribute is also used to identify objects and must be unique within its context. Objects may be referred to either by name or by alias. The name attribute often corresponds with how the object is referenced by other systems in the real world. The alias attribute provides an abstract identifier for an object. This allows building topologies which only need minimal changes when real-world objects are changed. Thus, for example, an interface referred to as “eth0” through a real-world device’s management console will is given a name value of “eth0” in a topology. The same interface is also given an alias value of “int0”. Throughout the topology the interface is referred to as “int0”. At a later time if the real-world interface is changed to “eth1” only the name attribute of the interface is changed; the rest of the topology is unaffected.

Additional attributes for each type of object are presented below. For completeness the name attribute is also listed.

### Topology

The following topology attributes are recognized by NTAF. All but name are optional.

| Name | Data Type | Single/Multi-valued | Description | Suggested Values |
|---|---|---|---|---|
| **name** | string | single | Optional unique identifier. Usually corresponds to real-world usage. | |
| **alias** | string | single | Optional abstract identifier | |
| **created** | datetime | single | The date (and possible time) the topology was created. | |
| **description** | string | single | A human readable description of the topology | |
| **type** | string | single | The type of topology. | |
| **user** | string | single | Person who created or owns this topology. | |

### Node

The following node attributes are recognized by NTAF. All but name are optional. There are also many attributes defined in Tables 2 and 3 of TS-004 for resources that may be used for nodes as needed.

| Name | Data Type | Single/Multi-valued | Description | Suggested Values |
|---|---|---|---|---|
| **name** | string | single | Mandatory unique identifier. Usually corresponds to real-world | |

| | | | | |
|---|---|---|---|---|
| | | | usage. | |
| **alias** | string | single | Optional abstract identifier | |
| **description** | string | single | A human readable description of this node | |
| **functionalClass** | string | multi | The functional classification of the node (i.e. the functions it primarily supports). A single node can provide multiple values. | Examples: router, ethernetSwitch, server, trafficGenerator. See Table 4 in TS-004 for complete list. |
| **hostName** | string | single | The host name of the node. | |
| **ip4Address** | string | single | The primary or management IP address of the node. Allowed format is IPv4 dotted decimal. Examples: 192.168.1.1, 10.210.255.200/32 | |
| **ip6Address** | string | single | The primary or management IP address of the node. Allowed format is IPv6 in hex colon separated hexadecimal. Examples: 2001:db8:85a3::8a2e:370:7334, ::1, 2001:db8:0:160::, abcd::192:168:8:1/126 | |
| **modelName** | string | single | Model name of the node (human readable). | |
| **physicalClass** | string | single | The physical classification of the resource (i.e. its physical form). Each resource provides the single value which most closely matches. | Examples: chassis, handheld, rackMountable,, virtual. See Table 3 in TS-004 for complete list. |

| | | | | |
|---|---|---|---|---|
| **productFamily** | string | single | Manufacturer supplied classification of the type of resource. | |
| **versionSoftware** | string | single | Identifies the version of the installed software of the resource (i.e. overall version). | |

### Interface

The following interface attributes are recognized by NTAF. All but name are optional. There are also many attributes defined in Table 6 of TS-004 for connectors that may be used for interfaces as needed.

| Name | Data Type | Single/Multi-valued | Description | Suggested Values |
|---|---|---|---|---|
| **name** | string | single | Mandatory unique identifier. Usually corresponds to real-world usage. | |
| **alias** | string | single | Optional abstract identifier | |
| **destination** | string | multi | Identifier of a connected interface. Identifier is of the form <node name>:<interface name>. Multiple values are allowed. | |
| **type** | string | single | The type of interface. | Examples: ethernet, ethernet100g, oc192, fiberChannel. For complete list see Table 7 of TS-004. |
| **ipv4Address** | string | single | Allowed format is IPv4 dotted decimal. Examples: 192.168.1.1/24, 10.210.255.200/32. Must include net mask in CIDR notation. | |

| | | | | |
|---|---|---|---|---|
| **ipv6Address** | string | multi | Allowed format is IPv6 in hex colon separated hexadecimal. Examples: 2001:db8:85a3::8a2e:370:7334/64 , 2001:db8:0:160::/64, abcd::192:168:8:1/126, fe80::3e94:d500:137:bdcf/64. Must include net mask. | |
| **isManagement** | boolean | single | If true this interface is a management port. | |
| **port** | integer | single | Port associated with ipAddress. | |
| **protocol** | string | multi | Protocol used to communicate (as management?) | |
| **managementProtocol** | string | multi | Protocol used to communicate | |
| **speed** | string | single | Communication speed of which this interface is capable | |
| **category** | string | single | The general category of the interface. | Examples: bundle, network. For complete list see category in Table 6 of TS-004. |
| **bundle** | string | single | Name of containing (parent) bundle interface. | |
| **vlanId** | string | single | Identifier of vlan with which this interface is associated | |

This standard adds values to the two of the lists provided in TS-004.

- The value "**ethernet**" is added to the list of interface (connector) types. It refers to generic ethernet which can then be distinguished using the speed attribute.
- The value "**bundle**" is added to the list of interface (connector) category values. It refers to interfaces which contain a bundle (or group) of other interfaces.

### Connection

The following connection attributes are recognized by NTAF and are mandatory.

| Name | Data Type | Single/Multi-valued | Description | Suggested Values |
|---|---|---|---|---|
| **name** | string | single | Mandatory unique identifier. Usually corresponds to real-world usage. | |
| **alias** | string | single | Optional abstract identifier | |
| **interface** | string | multi | Identifier of interfaces connected by connection. Each identifier is of the form <node name>:<interface name>. Multiple values are required. In other words there must be a minimum of two interfaces being connected. | |

# 6. Use Cases

## Basic Topology

This is a very basic example of a topology which defines two nodes (“New York” and “Los Angeles”). "New York" has a "west" interface which is connected to the "east" interface of "Los Angeles".

The following JSON represents the example topology with connections modeled using the destination attribute.

```
{
  "topology": {
    "nodes": [
      {
        "name": "New York",
        "description": "Times Square and more.",
        "interfaces": [{
            "name": "west",
            "destination": "Los Angeles:east",
            "type": "ethernet10g",
            "speed": "10g"
        }]
      },
      {
        "name": "Los Angeles",
        "description": "Hollywood, what more can we say?",
        "interfaces": [{
            "name": "east",
```

```
            "destination": "New York:west",
            "type": "ethernet10g",
            "speed": "10g"
        }]
      }
    ]
  }
}
```

The following JSON shows the same topology with the connections modeled as objects.

```
{
  "topology": {
    "nodes": [
      {
        "name": "New York",
        "description": "Times Square and more.",
        "interfaces": [{
            "name": "west",
            "type": "ethernet10g",
            "speed": "10g"
        }]
      },
      {
        "name": "Los Angeles",
        "description": "Hollywood, what more can we say?",
        "interfaces": [{
            "name": "east",
            "type": "ethernet10g",
            "speed": "10g"
        }]
      }
    ],
    "connections": [{
      "name": "redeye",
      "interfaces": ["New York:west", "Los Angeles:east"]
    }]
  }
}
```

The following XML represents the example topology with connections modeled using the destination attribute.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
  <nodes>
    <node>
      <description>Times Square and more.</description>
      <interfaces>
        <interface>
          <destinations>
            <destination>Los Angeles:east</destination>
          </destinations>
          <name>west</name>
          <speed>10g</speed>
```

```
          <type>ethernet10g</type>
        </interface>
      </interfaces>
      <name>New York</name>
    </node>
    <node>
      <description>Hollywood, what more can we say?</description>
      <interfaces>
        <interface>
          <destinations>
            <destination>New York:west</destination>
          </destinations>
          <name>east</name>
          <speed>10g</speed>
          <type>ethernet10g</type>
        </interface>
      </interfaces>
      <name>Los Angeles</name>
    </node>
  </nodes>
</topology>
```

The following XML shows the same topology with the connections modeled as objects.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
  <connections>
    <connection>
      <interfaces>
        <interface>New York:west</interface>
        <interface>Los Angeles:east</interface>
      </interfaces>
      <name>redeye</name>
    </connection>
  </connections>
  <nodes>
    <node>
      <description>Times Square and more.</description>
      <interfaces>
        <interface>
          <name>west</name>
          <speed>10g</speed>
          <type>ethernet10g</type>
        </interface>
      </interfaces>
      <name>New York</name>
    </node>
    <node>
      <description>Hollywood, what more can we say?</description>
      <interfaces>
        <interface>
          <name>east</name>
          <speed>10g</speed>
          <type>ethernet10g</type>
        </interface>
      </interfaces>
      <name>Los Angeles</name>
    </node>
  </nodes>
```

```
</topology>
```

## Sub-interfaces

Interfaces belonging to VLANs are also commonly used in networking and need to be represented in topologies. VLAN interfaces are represented using the vlanId attribute. A JSON example is given below.

```
{
  "topology": {
    "nodes": [
      {
        "name": "juniper",
        "functionalClasses": ["router"],
        "interfaces": [
          {
            "name": "eth0-1",
            "vlanId": "VLAN_0",
            "destinations": ["spirent:port1"],
            "ipAddressV4": "192.168.2.2/24",
            "category": "network",
            "type": "ethernet10g"
          },
          {
            "name": "eth0-2",
            "vlanId": "VLAN_1",
            "destinations": ["spirent:port2"],
            "ipAddressV4": "192.168.2.3/24",
            "category": "network",
            "type": "ethernet10g"
          }
        ]
      },
      {
        "name": "spirent",
        "functionalClasses": ["protocolEmulator", "trafficGenerator"],
        "interfaces": [
          {
            "name": "port1",
            "vlanId": "VLAN_0",
            "destinations": ["juniper:eth0-1"],
            "ipAddressV4": "192.168.2.5/24",
            "type": "ethernet10g"
          },
          {
            "name": "port2",
            "vlanId": "VLAN_1",
            "destinations": ["juniper:eth0-2"],
            "ipAddressV4": "192.168.2.6/24",
            "type": "ethernet10g"
          },
        ]
      }
    ]
  }
}
```

An XML example is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <nodes>
      <node>
         <functionalClasses>
           <functionalClass>router</functionalClass>
         </functionalClasses>
         <interfaces>
            <interface>
               <category>network</category>
               <destinations>
                 <destination>spirent:port1</destination>
               </destinations>
               <ipAddressV4>192.168.2.2/24</ipAddressV4>
               <name>eth0-1</name>
               <type>ethernet10g</type>
               <vlanId>VLAN_0</vlanId>
            </interface>
            <interface>
               <category>network</category>
               <destinations>
                 <destination>spirent:port2</destination>
               </destinations>
               <ipAddressV4>192.168.2.3/24</ipAddressV4>
               <name>eth0-2</name>
               <type>ethernet10g</type>
               <vlanId>VLAN_1</vlanId>
            </interface>
         </interfaces>
         <name>juniper</name>
      </node>
      <node>
         <functionalClasses>
            <functionalClass>protocolEmulator</functionalClass>
            <functionalClass>trafficGenerator</functionalClass>
         </functionalClasses>
         <interfaces>
            <interface>
               <destinations>
                 <destination>juniper:eth0-1</destination>
               </destinations>
               <ipAddressV4>192.168.2.5/24</ipAddressV4>
               <name>port1</name>
               <type>ethernet10g</type>
               <vlanId>VLAN_0</vlanId>
            </interface>
            <interface>
               <destinations>
                 <destination>juniper:eth0-2</destination>
               </destinations>
               <ipAddressV4>192.168.2.6/24</ipAddressV4>
               <name>port2</name>
               <type>ethernet10g</type>
               <vlanId>VLAN_1</vlanId>
            </interface>
         </interfaces>
         <name>spirent</name>
      </interface>
```

```
   </nodes>
</topology>
```

## Aggregating Interfaces

Link bundling is also common in networking, and likewise needs to be represented in topologies. The containing (parent) interface bundle has category of "bundle". Interfaces contained in a bundle have an attribute set to the name of the containing bundle interface. A JSON example is given below.

```
{
  "topology": {
    "nodes": [
      {
        "name": "juniper",
        "functionalClasses": ["router"],
        "interfaces": [
          {
            "name": "ae11",
            "destinations": ["spirent:mtv"],
            "ipAddressV4": "192.168.2.1/24",
            "speed": "10g",
            "category": "bundle",
          },
          {
            "name": "ae11-1",
            "bundle": "ae11",
            "destinations": ["spirent:port1"],
            "ipAddressV4": "192.168.2.2/24",
            "category": "network",
            "type": "ethernet10g"
          },
          {
            "name": "ae11-2",
            "destinations": ["spirent:port2"],
            "ipAddressV4": "192.168.2.3/24",
            "category": "network",
            "type": "ethernet10g"
          }
        ]
      },
      {
        "name": "spirent",
        "functionalClasses": ["protocolEmulator", "trafficGenerator"],
        "interfaces": [
          {
            "name": "mtv",
            "destinations": ["juniper:ae11"],
            "ipAddressV4": "192.168.2.4/24",
            "category": "bundle"
          },
          {
            "name": "port1",
            "bundle": "mtv",
            "destinations": ["juniper:ae11-1"],
            "ipAddressV4": "192.168.2.5/24",
            "type": "ethernet10g"
          },
          {
            "name": "port2",
            "bundle": "mtv",
            "destinations": ["juniper:ae11-2"],
```

```
            "ipAddressV4": "192.168.2.6/24",
            "type": "ethernet10g"
          },

        ]
      }
    ]
  }
}
```

An XML example is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <nodes>
      <node>
         <functionalClasses>
           <functionalClass>router</functionalClass>
         </functionalClasses>
         <interfaces>
            <interface>
               <category>bundle</category>
               <destinations>
                 <destination>spirent:mtv</destination>
               </destinations>
               <ipAddressV4>192.168.2.1/24</ipAddressV4>
               <name>ae11</name>
               <speed>10g</speed>
            </interface>
            <interface>
               <bundle>ae11</bundle>
               <category>network</category>
               <destinations>
                 <destination>spirent:port1</destination>
               </destinations>
               <ipAddressV4>192.168.2.2/24</ipAddressV4>
               <name>ae11-1</name>
               <type>ethernet10g</type>
            </interface>
            <interface>
               <category>network</category>
               <destinations>
                 <destination>spirent:port2</destination>
               </destinations>
               <ipAddressV4>192.168.2.3/24</ipAddressV4>
               <name>ae11-2</name>
               <type>ethernet10g</type>
            </interface>
         </interfaces>
         <name>juniper</name>
      </node>
      <node>
         <functionalClasses>
            <functionalClass>protocolEmulator</functionalClass>
            <functionalClass>trafficGenerator</functionalClass>
         </functionalClasses>
         <interfaces>
            <interface>
               <category>bundle</category>
               <destinations>
```

```
                <destination>juniper:ae11</destination>
              </destinations>
              <ipAddressV4>192.168.2.4/24</ipAddressV4>
              <name>mtv</name>
            </interface>
            <interface>
              <bundle>mtv</bundle>
              <destinations>
                <destination>juniper:ae11-1</destination>
              </destinations>
              <ipAddressV4>192.168.2.5/24</ipAddressV4>
              <name>port1</name>
              <type>ethernet10g</type>
            </interface>
            <interface>
              <bundle>mtv</bundle>
              <destinations>
                <destination>juniper:ae11-2</destination>
              </destinations>
              <ipAddressV4>192.168.2.6/24</ipAddressV4>
              <name>port2</name>
              <type>ethernet10g</type>
            </interface>
          </interfaces>
          <name>spirent</name>
        </node>
    </nodes>
</topology>
```

## Nested Interfaces

This use case also shows a simple topology with two connected routers. It shows how interfaces can be nested to represent logical interfaces using VLAN.

The JSON representation is given below.

```
{
  "topology": {
    "nodes": [
      {
        "name": "West",
        "description": "West router",
        "interfaces": [
          {
            "name": "west_link",
            "destinations": ["East:east_link"],
            "interfaces": [
              {
                "name": "west_link_0",
                "ipv4Address": "10.210.255.200/32",
                "ipv6Address": "abcd::10:210:255:200/128",
                "vlanId": "100",
                "destinations": ["East:east_link_0"]
              },
              {
                "name": "west_link_1",
                "ipv4Address": "10.211.255.200/32",
                "ipv6Address": "abcd::10:211:255:200/128",
                "vlanId": "101",
                "destinations": ["East:east_link_1"]
```

```
            }
          ]
        }
      ]
    },
    {
      "name": "East",
      "description": "East router",
      "interfaces": [
        {
          "name": "east_link",
          "destinations": ["West:west_link"],
          "interfaces": [
            {
              "name": "east_link_0",
              "ipv4Address": "10.210.255.201/32",
              "ipv6Address": "abcd::10:210:255:201/128",
              "vlanId": "100",
              "destinations": ["West:west_link_0"]
            },
            {
              "name": "east_link_1",
              "ipv4Address": "10.211.255.201/32",
              "ipv6Address": "abcd::10:211:255:201/128",
              "vlanId": "101",
              "destinations": ["West:west_link_1"]
            }
          ]
        }
      ]
    }
  ]
}
```

The XML representation is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <nodes>
      <node>
         <description>West router</description>
         <interfaces>
            <interface>
               <destinations>
                  <destination>East:east_link</destination>
               </destinations>
               <interfaces>
                  <interface>
                     <destinations>
                        <destination>East:east_link_0</destination>
                     </destinations>
                     <ipv4Address>10.210.255.200/32</ipv4Address>
                     <ipv6Address>abcd::10:210:255:200/128</ipv6Address>
                     <vlanId>100</vlanId>
                     <name>west_link_0</name>
                  </interface>
                  <interface>
                     <destinations>
                        <destination>East:east_link_1</destination>
```

```
                    </destinations>
                    <ipv4Address>10.211.255.200/32</ipv4Address>
                    <ipv6Address>abcd::10:211:255:200/128</ipv6Address>
                    <vlanId>101</vlanId>
                    <name>west_link_1</name>
                 </interface>
              </interfaces>
              <name>west_link</name>
           </interface>
        </interfaces>
        <name>West</name>
     </node>
     <node>
        <description>East router</description>
        <interfaces>
           <interface>
              <destinations>
                 <destination>West:west_link</destination>
              </destinations>
              <interfaces>
                 <interface>
                    <destinations>
                       <destination>West:west_link_0</destination>
                    </destinations>
                    <ipv4Address>10.210.255.201/32</ipv4Address>
                    <ipv6Address>abcd::10:210:255:201/128</ipv6Address>
                    <vlanId>100</vlanId>
                    <name>east_link_0</name>
                 </interface>
                 <interface>
                    <destinations>
                       <destination>West:west_link_1</destination>
                    </destinations>
                    <ipv4Address>10.211.255.201/32</ipv4Address>
                    <ipv6Address>abcd::10:211:255:201/128</ipv6Address>
                    <vlanId>101</vlanId>
                    <name>east_link_1</name>
                 </interface>
              </interfaces>
              <name>east_link</name>
           </interface>
        </interfaces>
        <name>East</name>
     </node>
  </nodes>
</topology>
```

## Nested Nodes

This use case shows a test tool connected to a router being tested. It shows how nested nodes can be used.

The JSON representation is given below.

```
{
  "topology": {
    "nodes": [
      {
```

```
    "name": "TestTool",
    "description": "Tool used to inject traffic and protocol data",
    "interfaces": [
      {
        "name": "port0",
        "destinations": ["DUT:eth0"]
      },
      {
        "name": "port1",
        "destinations": ["DUT:eth1"]
      }
    ],
    "nodes": [
      {
        "name": "emulated_router_0",
        "interfaces": [
          {
            "name": "eth0",
            "ipv4Address": "10.210.255.200/32",
            "ipv6Address": "abcd::10:210:255:200/128",
            "destinations": ["TestTool:port0"]
          }
        ]
      },
      {
        "name": "emulated_router_1",
        "interfaces": [
          {
            "name": "eth0",
            "ipv4Address": "10.211.255.200/32",
            "ipv6Address": "abcd::10:211:255:200/128",
            "destinations": ["TestTool:port1"]
          }
        ]
      }
    ]
  },
  {
    "name": "DUT",
    "description": "Physical router under test",
    "interfaces": [
      {
        "name": "eth0",
        "destinations": ["TestTool:port0"]
      },
      {
        "name": "eth1",
        "destinations": ["TestTool:port0"]
      }
    ],
    "nodes": [
      {
        "name": "logical_router_0",
        "interfaces": [
          {
            "name": "eth0",
            "ipv4Address": "10.210.255.201/32",
            "ipv6Address": "abcd::10:210:255:201/128",
            "destinations": ["DUT:eth0"]
          }
        ]
      },
```

```
          {
            "name": "logical_router_1",
            "interfaces": [
              {
                "name": "eth0",
                "ipv4Address": "10.211.255.201/32",
                "ipv6Address": "abcd::10:211:255:201/128",
                "destinations": ["DUT:eth1"]
              }
            ]
          }
        ]
      }
    ]
  }
}
```

The XML representation is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <nodes>
      <node>
         <description>Tool used to inject traffic and protocol data</description>
         <interfaces>
            <interface>
               <destinations>
                  <destination>DUT:eth0</destination>
               </destinations>
               <name>port0</name>
            </interface>
            <interface>
               <destinations>
                  <destination>DUT:eth1</destination>
               </destinations>
               <name>port1</name>
            </interface>
         </interfaces>
         <name>TestTool</name>
         <nodes>
            <node>
               <interfaces>
                  <interface>
                     <destinations>
                        <destination>TestTool:port0</destination>
                     </destinations>
                     <ipv4Address>10.210.255.200/32</ipv4Address>
                     <ipv6Address>abcd::10:210:255:200/128</ipv6Address>
                     <name>eth0</name>
                  </interface>
               </interfaces>
               <name>emulated_router_0</name>
            </node>
            <node>
               <interfaces>
                  <interface>
                     <destinations>
                        <destination>TestTool:port1</destination>
                     </destinations>
                     <ipv4Address>10.211.255.200/32</ipv4Address>
```

```
                  <ipv6Address>abcd::10:211:255:200/128</ipv6Address>
                  <name>eth0</name>
               </interface>
            </interfaces>
            <name>emulated_router_1</name>
         </node>
      </nodes>
   </node>
   <node>
      <description>Physical router under test</description>
      <interfaces>
         <interface>
            <destinations>
               <destination>TestTool:port0</destination>
            </destinations>
            <name>eth0</name>
         </interface>
         <interface>
            <destinations>
               <destination>TestTool:port0</destination>
            </destinations>
            <name>eth1</name>
         </interface>
      </interfaces>
      <name>DUT</name>
      <nodes>
         <node>
            <interfaces>
               <interface>
                  <destinations>
                     <destination>DUT:eth0</destination>
                  </destinations>
                  <ipv4Address>10.210.255.201/32</ipv4Address>
                  <ipv6Address>abcd::10:210:255:201/128</ipv6Address>
                  <name>eth0</name>
               </interface>
            </interfaces>
            <name>logical_router_0</name>
         </node>
         <node>
            <interfaces>
               <interface>
                  <destinations>
                     <destination>DUT:eth1</destination>
                  </destinations>
                  <ipv4Address>10.211.255.201/32</ipv4Address>
                  <ipv6Address>abcd::10:211:255:201/128</ipv6Address>
                  <name>eth0</name>
               </interface>
            </interfaces>
            <name>logical_router_1</name>
         </node>
      </nodes>
   </node>
 </nodes>
</topology>
```

## Aliases and Management Ports

This use case shows a simple topology with two connected routers. It shows hows aliases can be used as abstract identifiers. It also shows how management ports are represented.

The JSON representation is given below.

```
{
  "topology": {
    "nodes": [
      {
        "name": "72k-206-235-1",
        "alias": "rtr1",
        "modelName": "7206vxr",
        "interfaces": [
          {
            "managementProtocol": "telnet",
            "ipAddress": "10.85.69.5",
            "port": "2001",
            "isManagement": "true"
          },
          {
            "managementProtocol": "telnet",
            "ipAddress": "10.85.69.5",
            "port": "2001",
            "isManagement": "true"
          },
          {
              "name": "fa2/0",
              "alias": "int10",
              "destinations": ["rtr12:int1"],
              "category": "network",
              "type": "ethernetFast"
          },
          {
            "name": "po3/0",
            "alias": "int11",
            "destinations": ["rtr12:int9"],
            "category": "network",
            "type": "oc3"
          }
        ]
      },
      {
        "name": "ASR1k-206-234-3",
        "alias": "rtr12",
        "modelName": "asr1006",
        "interfaces": [
          {
            "protocol": "telnet",
            "ipAddress": "10.85.69.4",
            "port": "2005",
            "name": "Active_console",
            "isManagement": "true"
          },
          {
            "protocol": "telnet",
            "ipAddress": "10.85.69.4",
            "port": "2006",
            "name": "Standby_console",
```

```
            "isManagement": "true"
          },
          {
            "name": "gi0",
            "alias": "tftp_int",
            "protocol": "telnet",
            "ipAddress": "5.31.2.22",
            "mask": "255.255.0.0",
            "category": "network",
            "type": "ethernetGigabit"
          },
          {
            "name": "fa0/0/0",
            "alias": "int1",
            "destinations": ["rtr1:int10"],
            "category": "network",
            "type": "ethernetFast"
          },
          {
            "name": "po0/1/0",
            "alias": "int9",
            "destinations": ["rtr1:int11"],
            "category": "network",
            "type": "oc3"
          }
        ]
      }
    ]
  }
}
```

The XML representation is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
  <nodes>
    <node>
      <alias>rtr1</alias>
      <interfaces>
        <interface>
          <ipAddress>10.85.69.5</ipAddress>
          <port>2001</port>
          <protocol>telnet</protocol>
          <isManagement>telnet</isManagement>
        </interface>
        <interface>
          <alias>int10</alias>
          <category>network</category>
          <destinations>
            <destination>rtr12:int1</destination>
          </destinations>
          <name>fa2/0</name>
          <type>ethernetFast</type>
        </interface>
        <interface>
          <alias>int11</alias>
          <category>network</category>
          <destinations>
            <destination>rtr12:int9</destination>
          </destinations>
```

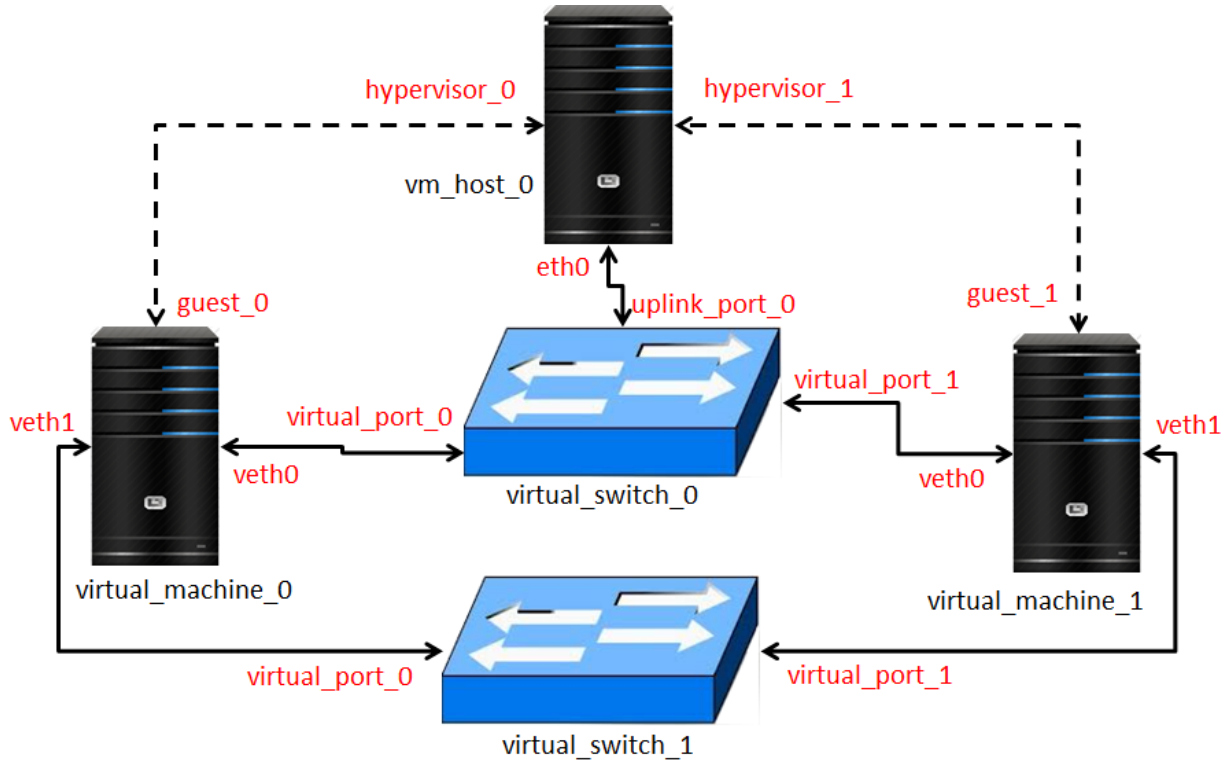
```
          <name>po3/0</name>
          <type>oc3</type>
        </interface>
      </interfaces>
      <modelName>7206vxr</modelName>
      <name>72k-206-235-1</name>
    </node>
    <node>
      <alias>rtr12</alias>
      <interfaces>
        <interface>
          <ipAddress>10.85.69.4</ipAddress>
          <name>Active_console</name>
          <port>2005</port>
          <managementProtocol>telnet</managementProtocol>
          <isManagement>true</isManagement>
        </interface>
        <interface>
          <ipAddress>10.85.69.4</ipAddress>
          <name>Standby_console</name>
          <port>2006</port>
          <managementProtocol>telnet</managementProtocol>
          <isManagement>true</isManagement>
        </interface>
        <interface>
          <alias>tftp_int</alias>
          <category>network</category>
          <ipAddress>5.31.2.22</ipAddress>
          <mask>255.255.0.0</mask>
          <name>gi0</name>
          <protocol>telnet</protocol>
          <type>ethernetGigabit</type>
        </interface>
        <interface>
          <alias>int1</alias>
          <category>network</category>
          <destinations>
            <destination>rtr1:int10</destination>
          </destinations>
          <name>fa0/0/0</name>
          <type>ethernetFast</type>
        </interface>
        <interface>
          <alias>int9</alias>
          <category>network</category>
          <destinations>
            <destination>rtr1:int11</destination>
          </destinations>
          <name>po0/1/0</name>
          <type>oc3</type>
        </interface>
      </interfaces>
      <modelName>asr1006</modelName>
      <name>ASR1k-206-234-3</name>
    </node>
  </nodes>
</topology>
```

## Virtual Machine Topology

This use case is derived from the Virtual Machine Connectors use case in TS-004. It models a virtualized environment. There is a host machine (hypervisor) which has two virtual switches (one internal, one external) and two virtual machines installed. Each virtual machine has two virtual Ethernet adapters which are connected to the two virtual switches. One of the defined virtual switches (external) has an uplink via a physical port.

The network topology is shown graphically below.




The JSON representation is given below.

```
{
  "topology": {
    "nodes": [
      {
        "name": "vm_host_0",
        "description": "Host for virtual machines.",
        "functionalClasses": ["hypervisor"],
        "physicalClass": "chassis",
        "interfaces": [
          {
            "name": "eth0",
            "destinations": ["virtual_switch_0:uplink_port_0"],
            "category": "network",
            "type": "ethernetGigabit",
            "speed": "100Mbs"
          },
          {
            "name": "hypervisor_0",
```

```
        "destinations": ["virtual_machine_0:guest_0", "virtual_machine_1:guest_1"],
        "category": "installation",
        "type": "software"
      }
    ]
  },
  {
    "name": "virtual_switch_0",
    "description": "Bridge to physical network",
    "functionalClasses": ["ethernetSwitch"],
    "physicalClass": "virtual",
    "interfaces": [
      {
        "name": "uplink_port_0",
        "destinations": ["vm_host_0:eth0"],
        "category": "network",
        "type": "ethernetGigabit",
        "speed": "100Mbs"
      },
      {
        "name": "virtual_port_0",
        "destinations": ["virtual_machine_0:veth0"],
        "category": "network",
        "type": "ethernetGigabit",
        "speed": "100Mbs"
      },
      {
        "name": "virtual_port_1",
        "destinations": ["virtual_machine_1:veth0"],
        "category": "network",
        "type": "ethernetGigabit",
        "speed": "100Mbs"
      }
    ]
  },
  {
    "name": "virtual_switch_1",
    "description": "Internal VM network",
    "functionalClasses": ["ethernetSwitch"],
    "physicalClass": "virtual",
    "interfaces": [
      {
        "name": "virtual_port_0",
        "destinations": ["virtual_machine_0:veth1"],
        "category": "network",
        "type": "ethernetGigabit",
        "speed": "100Mbs"
      },
      {
        "name": "virtual_port_1",
        "destinations": ["virtual_machine_1:veth1"],
        "category": "network",
        "type": "ethernetGigabit",
        "speed": "100Mbs"
      }
    ]
  },
  {
    "name": "virtual_machine_0",
    "description": "First virtual machine",
    "functionalClasses": ["server"],
    "physicalClass": "virtual",
```

```
        "interfaces": [
          {
            "name": "veth0",
            "destinations": ["virtual_switch_0:virtual_port_0"],
            "category": "network",
            "type": "ethernetGigabit",
            "speed": "100Mbs"
          },
          {
            "name": "veth1",
            "destinations": ["virtual_switch_1:virtual_port_0"],
            "category": "network",
            "type": "ethernetGigabit",
            "speed": "100Mbs"
          }
        ]
      },
      {
        "name": "virtual_machine_1",
        "description": "First virtual machine",
        "functionalClasses": ["server"],
        "physicalClass": "virtual",
        "interfaces": [
          {
            "name": "veth0",
            "destinations": ["virtual_switch_0:virtual_port_1"],
            "category": "network",
            "type": "ethernetGigabit",
            "speed": "100Mbs"
          },
          {
            "name": "veth1",
            "destinations": ["virtual_switch_1:virtual_port_1"],
            "category": "network",
            "type": "ethernetGigabit",
            "speed": "100Mbs"
          }
        ]
      }
    ]
  }
}
```

The XML representation is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
  <nodes>
    <node>
      <description>Host for virtual machines.</description>
      <functionalClasses>
        <functionalClass>hypervisor</functionalClass>
      </functionalClasses>
      <interfaces>
        <interface>
          <category>network</category>
          <destinations>
            <destination>virtual_switch_0:uplink_port_0</destination>
          </destinations>
          <name>eth0</name>
          <speed>100Mbs</speed>
```

```
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>installation</category>
        <destinations>
          <destination>virtual_machine_0:guest_0</destination>
          <destination>virtual_machine_1:guest_1</destination>
        </destinations>
        <name>hypervisor_0</name>
        <type>software</type>
      </interface>
    </interfaces>
    <name>vm_host_0</name>
    <physicalClass>chassis</physicalClass>
  </node>
  <node>
    <description>Bridge to physical network</description>
    <functionalClasses>
      <functionalClass>ethernetSwitch</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>vm_host_0:eth0</destination>
        </destinations>
        <name>uplink_port_0</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_machine_0:veth0</destination>
        </destinations>
        <name>virtual_port_0</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_machine_1:veth0</destination>
        </destinations>
        <name>virtual_port_1</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>virtual_switch_0</name>
    <physicalClass>virtual</physicalClass>
  </node>
  <node>
    <description>Internal VM network</description>
    <functionalClass>ethernetSwitch</functionalClass>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_machine_0:veth1</destination>
        </destinations>
        <name>virtual_port_0</name>
```

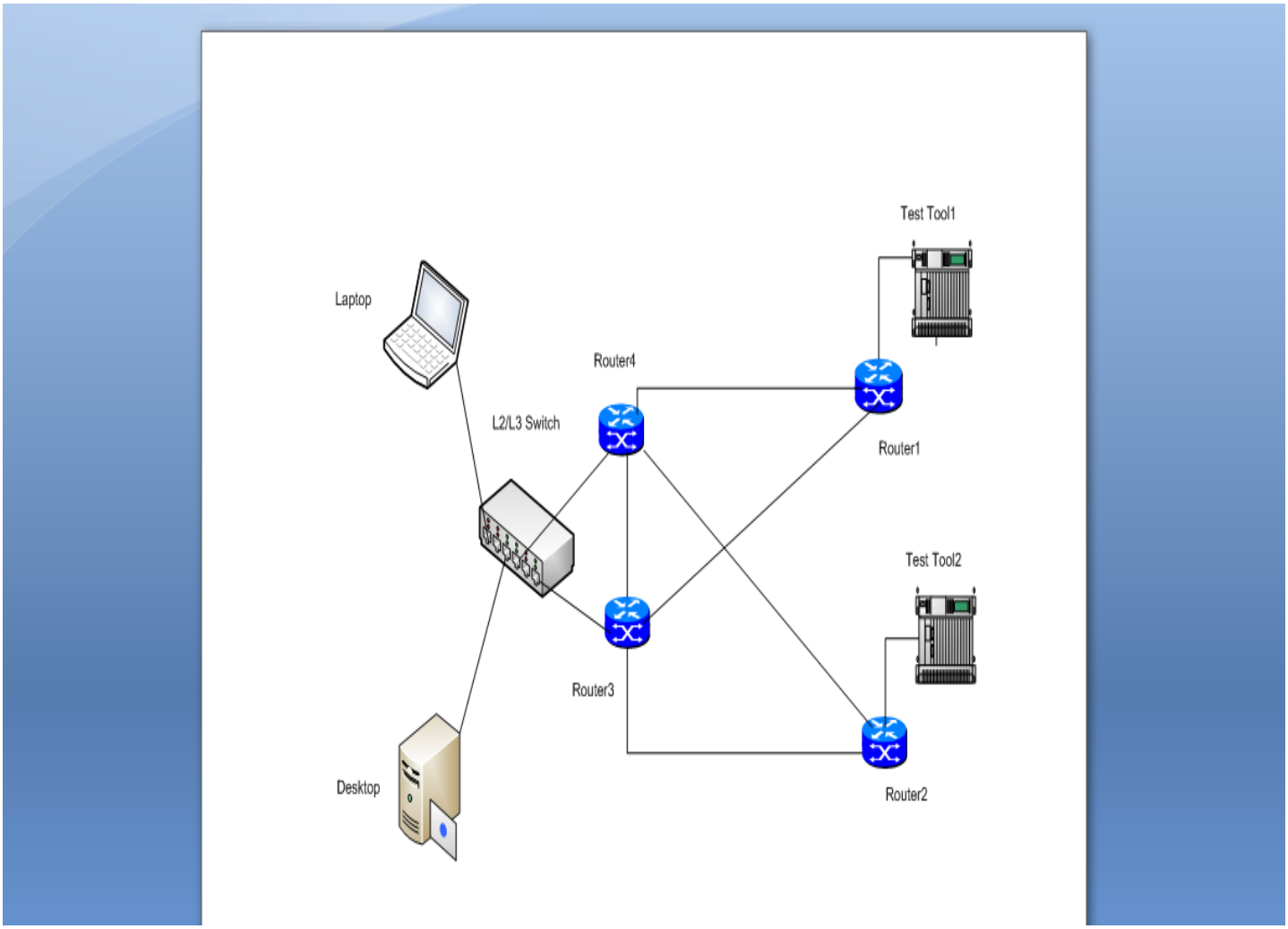
```
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_machine_1:veth1</destination>
        </destinations>
        <name>virtual_port_1</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>virtual_switch_1</name>
    <physicalClass>virtual</physicalClass>
  </node>
  <node>
    <description>First virtual machine</description>
    <functionalClasses>
      <functionalClass>server</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_switch_0:virtual_port_0</destination>
        </destinations>
        <name>veth0</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_switch_1:virtual_port_0</destination>
        </destinations>
        <name>veth1</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>virtual_machine_0</name>
    <physicalClass>virtual</physicalClass>
  </node>
  <node>
    <description>First virtual machine</description>
    <functionalClasses>
      <functionalClass>server</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>virtual_switch_0:virtual_port_1</destination>
        </destinations>
        <name>veth0</name>
        <speed>100Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
```

```
            <destination>virtual_switch_1:virtual_port_1</destination>
          </destinations>
          <name>veth1</name>
          <speed>100Mbs</speed>
          <type>ethernetGigabit</type>
        </interface>
      </interfaces>
      <name>virtual_machine_1</name>
      <physicalClass>virtual</physicalClass>
    </node>
  </nodes>
</topology>
```

## Physical Machine Topology

This use case models a physical test environment that includes 4 routers, layer2/3 switch, test tool, laptop and desktop. These systems are physically connected to create a network environment where layer2 to layer7 traffic can flow across the network.



The JSON representation is given below.

```
{
  "topology": {
    "nodes":[
      {
        "name":"router1",
        "description":"Serving MPLS Core.",
        "functionalClasses":["router"],
        "physicalClass":"chassis",
        "interfaces":[
          {
            "name":"ge-0/0/0",
            "destinations":["router3:ge-0/0/0"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          },
          {
            "name":"ge-1/1/1",
            "destinations":["router4:ge-1/1/1"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          },
          {
            "name":"xe-10/0/0",
            "destinations":["Test Tool1:port1"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"10000Mbs"
          }
        ]
      },
      {
        "name":"router2",
        "description":"Serving MPLS Core.",
        "functionalClasses":["router"],
        "physicalClass":"chassis",
        "interfaces":[
          {
            "name":"ge-1/1/1",
            "destinations":["router3:ge-1/1/1"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          },
          {
            "name":"ge-2/2/1",
            "destinations":["router4:ge-2/2/1"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          },
          {
            "name":"xe-10/1/1",
```

```
        "destinations":["Test Tool2:port1"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"10000Mbs"
      }
    ]
  },
  {
    "name":"router3",
    "description":"Serving Edge Network.",
    "functionalClasses":["router"],
    "physicalClass":"chassis",
    "interfaces":[
      {
        "name":"ge-0/0/0",
        "destinations":["router1:ge-0/0/0"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"ge-1/1/1",
        "destinations":["router2:ge-1/1/1"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"ge-2/2/2",
        "destinations":["L2/L3 Switch:Eth1"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      }
    ]
  },
  {
    "name":"router4",
    "description":"Serving Edge Network.",
    "functionalClasses":["router"],
    "physicalClass":"chassis",
    "interfaces":[
      {
        "name":"ge-1/1/1",
        "destinations":["router1:ge-1/1/1"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"ge-2/2/1",
        "destinations":["router2:ge-2/2/1"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"ge-2/2/2",
        "destinations":["L2/L3 Switch:Eth2"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
```

```
      }
    ]
  },
  {
    "name":"L2/L3 Switch",
    "description":"Access Switch",
    "functionalClasses":["Switch"],
    "physicalClass":"chassis",
    "interfaces":[
      {
        "name":"Eth1",
        "destinations":["router3:ge-2/2/2"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"Eth2",
        "destinations":["router4:ge-2/2/2"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"Eth3",
        "destinations":["laptop:eth0"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      },
      {
        "name":"Eth4",
        "destinations":["desktop:eht0"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      }
    ]
  },
  {
    "name":"laptop",
    "physicalClass":"laptop",
    "interfaces":[
      {
        "name":"eth0",
        "destinations":["L2/L3 Switch:Eth3"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
      }
    ]
  },
  {
    "name":"desktop",
    "physicalClass":"desktop",
    "interfaces":[
      {
        "name":"eth0",
        "destinations":["L2/L3 Switch:Eth4"],
        "category":"network",
        "type":"ethernetGigabit",
        "speed":"1000Mbs"
```

```
          }
        ]
      },
      {
        "name":"Test Tool1",
        "physicalClass":"rackMountable",
        "interfaces":[
          {
            "name":"port1",
            "destinations":["router1:xe-10/0/0"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          }
        ]
      },
      {
        "name":"Test Tool2",
        "physicalClass":"rackMountable",
        "interfaces":[
          {
            "name":"port1",
            "destinations":["router2:xe-10/1/1"],
            "category":"network",
            "type":"ethernetGigabit",
            "speed":"1000Mbs"
          }
        ]
      }
    ]
  }
}
```

The XML representation is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
  <nodes>
    <node>
      <description>Serving MPLS Core.</description>
      <functionalClasses>
        <functionalClass>router</functionalClass>
      </functionalClasses>
      <interfaces>
        <interface>
          <category>network</category>
          <destinations>
            <destination>router3:ge-0/0/0</destination>
          </destinations>
          <name>ge-0/0/0</name>
          <speed>1000Mbs</speed>
          <type>ethernetGigabit</type>
        </interface>
        <interface>
          <category>network</category>
          <destinations>
            <destination>router4:ge-1/1/1</destination>
          </destinations>
          <name>ge-1/1/1</name>
          <speed>1000Mbs</speed>
          <type>ethernetGigabit</type>
```

```
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>Test Tool1:port1</destination>
        </destinations>
        <name>xe-10/0/0</name>
        <speed>10000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>router1</name>
    <physicalClass>chassis</physicalClass>
  </node>
  <node>
    <description>Serving MPLS Core.</description>
    <functionalClasses>
      <functionalClass>router</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router3:ge-1/1/1</destination>
        </destinations>
        <name>ge-1/1/1</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router4:ge-2/2/1</destination>
        </destinations>
        <name>ge-2/2/1</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>Test Tool2:port1</destination>
        </destinations>
        <name>xe-10/1/1</name>
        <speed>10000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>router2</name>
    <physicalClass>chassis</physicalClass>
  </node>
  <node>
    <description>Serving Edge Network.</description>
    <functionalClasses>
      <functionalClass>router</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router1:ge-0/0/0</destination>
        </destinations>
```

```
        <name>ge-0/0/0</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router2:ge-1/1/1</destination>
        </destinations>
        <name>ge-1/1/1</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>L2/L3 Switch:Eth1</destination>
        </destinations>
        <name>ge-2/2/2</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>router3</name>
    <physicalClass>chassis</physicalClass>
  </node>
  <node>
    <description>Serving Edge Network.</description>
    <functionalClasses>
      <functionalClass>router</functionalClass>
    </functionalClasses>
    <interfaces>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router1:ge-1/1/1</destination>
        </destinations>
        <name>ge-1/1/1</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>router2:ge-2/2/1</destination>
        </destinations>
        <name>ge-2/2/1</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
      <interface>
        <category>network</category>
        <destinations>
          <destination>L2/L3 Switch:Eth2</destination>
        </destinations>
        <name>ge-2/2/2</name>
        <speed>1000Mbs</speed>
        <type>ethernetGigabit</type>
      </interface>
    </interfaces>
    <name>router4</name>
    <physicalClass>chassis</physicalClass>
```

```
</node>
<node>
  <description>Access Switch</description>
  <functionalClasses>
    <functionalClass>Switch</functionalClass>
  </functionalClasses>
  <interfaces>
    <interface>
      <category>network</category>
      <destinations>
        <destination>router3:ge-2/2/2</destination>
      </destinations>
      <name>Eth1</name>
      <speed>1000Mbs</speed>
      <type>ethernetGigabit</type>
    </interface>
    <interface>
      <category>network</category>
      <destinations>
        <destination>router4:ge-2/2/2</destination>
      </destinations>
      <name>Eth2</name>
      <speed>1000Mbs</speed>
      <type>ethernetGigabit</type>
    </interface>
    <interface>
      <category>network</category>
      <destinations>
        <destination>laptop:eth0</destination>
      </destinations>
      <name>Eth3</name>
      <speed>1000Mbs</speed>
      <type>ethernetGigabit</type>
    </interface>
    <interface>
      <category>network</category>
      <destinations>
        <destination>desktop:eht0</destination>
      </destinations>
      <name>Eth4</name>
      <speed>1000Mbs</speed>
      <type>ethernetGigabit</type>
    </interface>
  </interfaces>
  <name>L2/L3 Switch</name>
  <physicalClass>chassis</physicalClass>
</node>
<node>
  <interfaces>
    <interface>
      <category>network</category>
      <destinations>
        <destination>L2/L3 Switch:Eth3</description>
      </destinations>
      <name>eth0</name>
      <speed>1000Mbs</speed>
      <type>ethernetGigabit</type>
    </interface>
  </interfaces>
  <name>laptop</name>
  <physicalClass>laptop</physicalClass>
</node>
```

```
    <node>
      <interfaces>
        <interface>
          <category>network</category>
          <destinations>
            <destination>L2/L3 Switch:Eth4</destination>
          </destinations>
          <name>eth0</name>
          <speed>1000Mbs</speed>
          <type>ethernetGigabit</type>
        </interface>
      </interfaces>
      <name>desktop</name>
      <physicalClass>desktop</physicalClass>
    </node>
    <node>
      <interfaces>
        <interface>
          <category>network</category>
          <destinations>
            <destination>router1:xe-10/0/0</destination>
          </destinations>
          <name>port1</name>
          <speed>1000Mbs</speed>
          <type>ethernetGigabit</type>
        </interface>
      </interfaces>
      <name>Test Tool1</name>
      <physicalClass>rackMountable</physicalClass>
    </node>
    <node>
      <interfaces>
        <interface>
          <category>network</category>
          <destinations>
            <destination>router2:xe-10/1/1</destination>
          </destinations>
          <name>port1</name>
          <speed>1000Mbs</speed>
          <type>ethernetGigabit</type>
        </interface>
      </interfaces>
      <name>Test Tool2</name>
      <physicalClass>rackMountable</physicalClass>
    </node>
  </nodes>
</topology>
```

## 7. Extensions

Topologies may define additional attributes for any objects. Such additional attributes must be named according to the principles laid out in Section 3 of TS-004 under "Additional Information". Briefly, attribute names must indicate attribute origin.

# 8. Graphical Representations

This document does not attempt to define how topologies should be represented graphically. However, it is expected that software tools will take advantage of this specification to draw pictorial representations of topologies.