TEAS Working Group
Internet Draft
Intended status: Standards Track

Xufeng Liu
Ericsson
Igor Bryskin
ADVA Optical Networking
Vishnu Pavan Beeram
Juniper Networks
Tarek Saad
Cisco Systems Inc
Himanshu Shah
Ciena
Oscar Gonzalez De Dios
Telefonica

Expires: April 19, 2016 October 19, 2015

YANG Data Model for TE Topologies
draft-ietf-teas-yang-te-topo-02

Status of this Memo

Copyright Notice

Abstract

This document defines a YANG data model for representing, retrieving and manipulating TE Topologies. The model serves as a base model that other technology specific TE Topology models can augment.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

Table of Contents

1. Introduction

The Traffic Engineering Database (TED) is an essential component of Traffic Engineered (TE) systems that are based on MPLS-TE [RFC2702] and GMPLS [RFC3945]. The TED is a collection of all TE information about all TE nodes and TE links in the network. The TE Topology is a schematic arrangement of TE nodes and TE links present in a given TED. There could be one or more TE Topologies present in a given Traffic Engineered system. The TE Topology is the topology on which path computational algorithms are run to compute Traffic Engineered Paths (TE Paths).

This document defines a YANG [RFC6020] data model for representing and manipulating TE Topologies. This model contains technology agnostic TE Topology building blocks that can be augmented and used by other technology-specific TE Topology models.

1.1. Terminology

TED: The Traffic Engineering Database is a collection of all TE information about all TE nodes and TE links in a given network.

TE-Topology: The TE Topology is a schematic arrangement of TE nodes and TE links in a given TED. It forms the basis for a graph suitable for TE path computations.

Native TE Topology: Native TE Topology is a topology that is native to a given provider network. This is the topology on which path computational algorithms are run to compute TE Paths.

Customized TE Topology: Customized TE Topology is a custom topology that is produced by a provider for a given Client. This topology

typically augments the Client's Native TE Topology. Path computational algorithms aren't typically run on the Customized TE Topology; they are run on the Client's augmented Native TE Topology.

1.2. Tree Structure - Legend

A simplified graphical representation of the data model is presented in Section 5 of this document. The following notations are used for the YANG model data tree representation.

```
   <status> <flags> <name> <opts> <type>

     <status> is one of:
       +  for current
       x  for deprecated
       o  for obsolete

     <flags> is one of:
       rw  for read-write configuration data
       ro  for read-only non-configuration data
       -x  for execution rpcs
       -n  for notifications

     <name> is the name of the node

      If the node is augmented into the tree from another module,
      its name is printed as <prefix>:<name>

     <opts> is one of:
       ? for an optional leaf or node
       ! for a presence container
       * for a leaf-list or list
       Brackets [<keys>] for a list's keys
       Curly braces {<condition>} for optional feature that make
       node conditional

       Colon : for marking case nodes
       Ellipses ("...") subtree contents not shown

       Parentheses enclose choice and case nodes, and case nodes are
       also marked with a colon (":").

     <type> is the name of the type for leafs and leaf-lists.
```

1.3. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are prefixed using the standard prefix associated with the corresponding YANG imported modules, as shown in Table 1.

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and corresponding YANG modules

2. Characterizing TE Topologies

The data model proposed by this document takes the following characteristics of TE Topologies into account:

- TE Topology is an abstract control-plane representation of the data-plane topology. Hence attributes specific to the data-plane must make their way into the corresponding TE Topology modeling. The TE Topology comprises of dynamic auto-discovered data (data that may change frequently - example: unreserved bandwidth available on data-plane links) as well as fairly static data (data that rarely changes- examples: layer network identification, switching and adaptation capabilities and limitations, fate sharing, administrative colors) associated with data-plane nodes and links. It is possible for a single TE Topology to encompass TE information at multiple switching layers.

- TE Topologies are protocol independent. Information about topological elements may be learnt via link-state protocols, but the topology can exist without being dependent on any particular protocol.

- TE Topology may not be congruent to the routing topology (topology constructed based on routing adjacencies) in a given TE System. There isn't always a one-to-one association between a TE-link and a routing adjacency. For example, the presence of a TE link between a pair of nodes doesn't necessarily imply the existence of a routing-adjacency between these nodes.

- Each TE Topological element has an information source associated with it. In some scenarios, there could be more than one information source associated with each topological element.

- TE Topologies can be hierarchical. Each node and link of a given TE Topology can be associated with respective underlay topology. This means that each node and link of a given TE Topology can be associated with an independent stack of supporting TE Topologies.

- TE Topologies can be customized. TE topologies of a given network presented by the network provider to its client could be customized on per-client request basis. This customization could be performed by provider, by client or by provider/client negotiation. The relationship between a customized topology (as presented to the client) and provider's native topology (as known in its entirety to the provider itself) could be captured as hierarchical (overlay-underlay), but otherwise the two topologies are decoupled from each other.

3. Model Applicability

3.1. Native TE Topologies

The model discussed in this draft can be used to represent and retrieve native TE topologies on a given TE system.

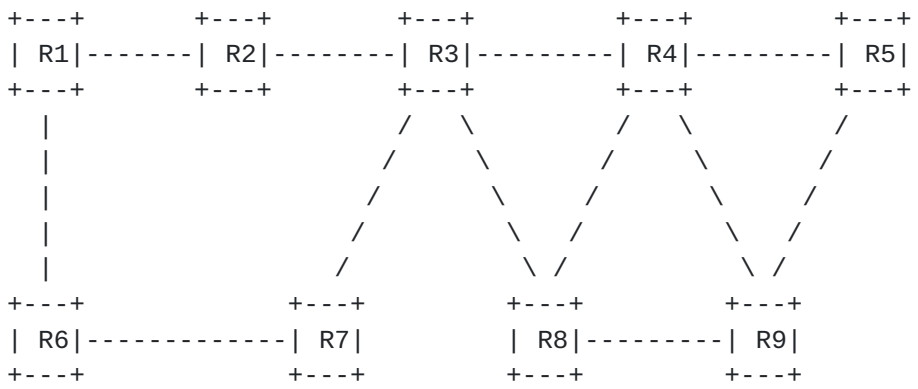

Figure 1a: Example Network Topology

Consider the network topology depicted in Figure 1a (R1 .. R9 are nodes representing routers). An implementation MAY choose to construct a native TE Topology using all nodes and links present in the given TED as depicted in Figure 1b. The data model proposed in this document can be used to retrieve/represent this TE topology.

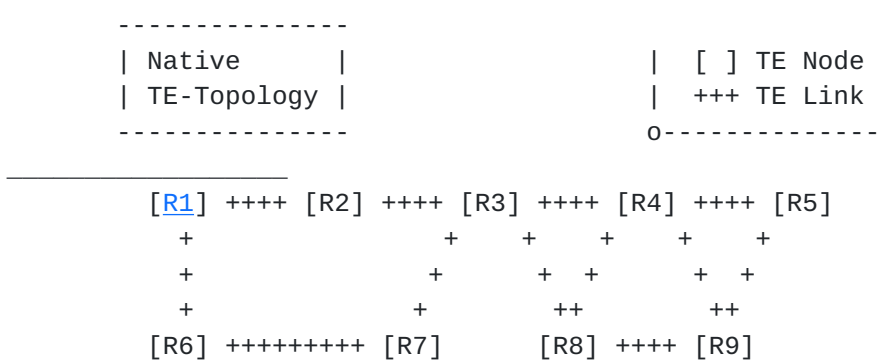

Figure 1b: Native TE Topology as seen on Node R3

Consider the case of the topology being split in a way that some nodes participate in OSPF-TE while others participate in ISIS-TE (Figure 2a). An implementation MAY choose to construct separate TE Topologies based on the information source. The native TE Topologies constructed using only nodes and links that were learnt via a specific information source are depicted in Figure 2b. The data model proposed in this document can be used to retrieve/represent these TE topologies.

Similarly, the data model can be used to represent/retrieve a TE Topology that is constructed using only nodes and links that belong to a particular technology layer. The data model is flexible enough to retrieve and represent many such native TE Topologies.

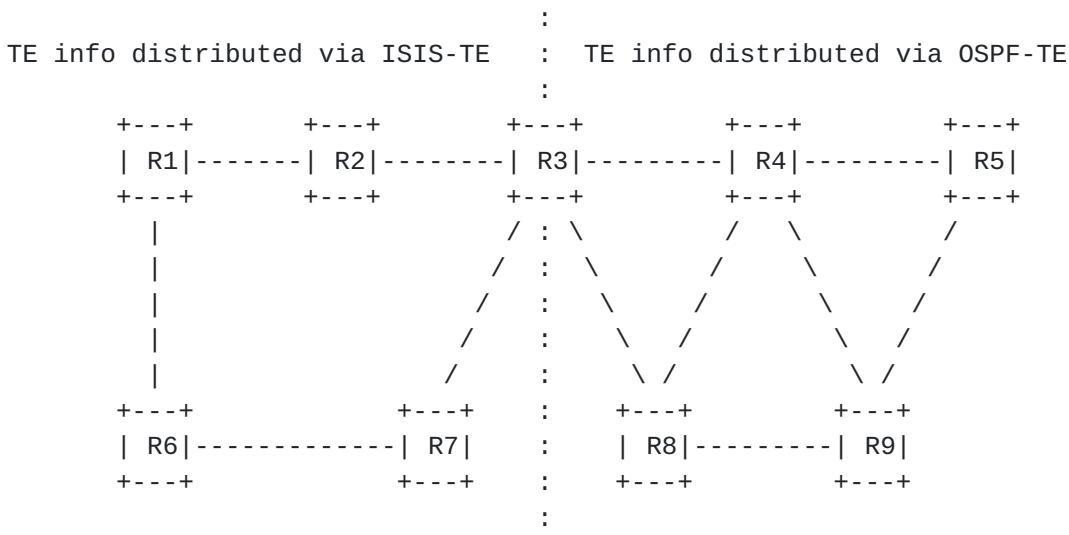

Figure 2a: Example Network Topology

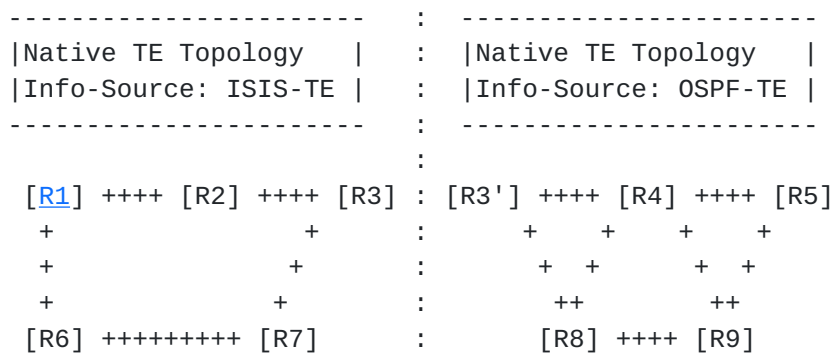

Figure 2b: Native TE Topologies as seen on Node R3

3.2. Customized TE Topologies

The model discussed in this draft can be used to represent, retrieve and manipulate customized TE Topologies. The model allows the provider to present the network in abstract TE Terms on a per client basis. These customized topologies contain sufficient information for the path computing client to select paths according to its policies.

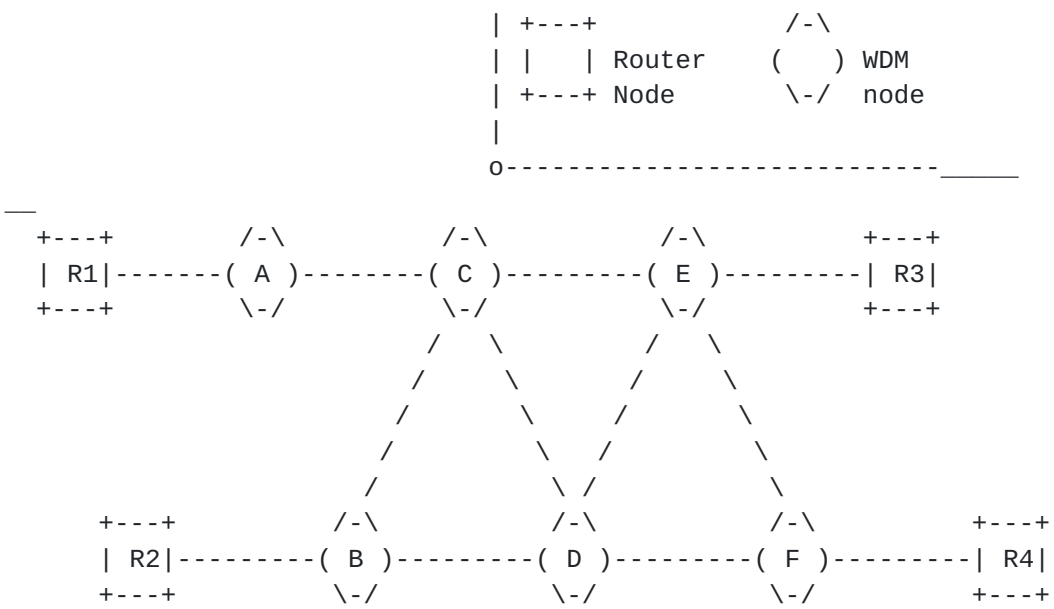

Figure 3: Example packet optical topology

Consider the network topology depicted in Figure 3. This is a typical packet optical transport deployment scenario where the WDM layer network domain serves as a Server Network Domain providing transport connectivity to the packet layer network Domain (Client Network Domain). Nodes R1, R2, R3 and R4 are IP routers that are connected to an Optical WDM transport network. A, B, C, D, E and F are WDM nodes that constitute the Server Network Domain.

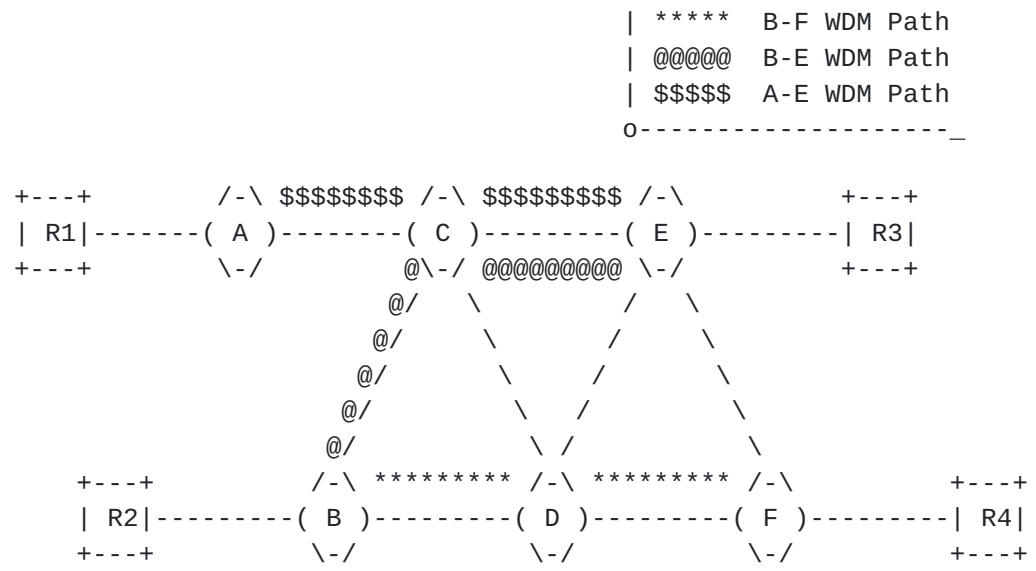

Figure 4a: Paths within the provider domain

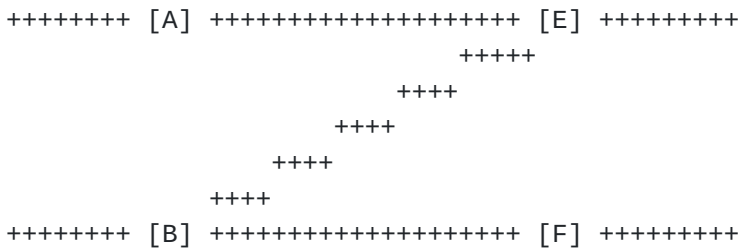

Figure 4b: Customized TE Topology provided to the Client

The goal here is to augment the Client TE Topology with a customized TE Topology provided by the WDM network. Given the availability of the paths A-E, B-F and B-E (Figure 4a), a customized TE Topology as depicted in Figure 4b is provided to the Client. This customized TE Topology is merged with the Client's Native TE Topology and the resulting topology is depicted in Figure 4c.

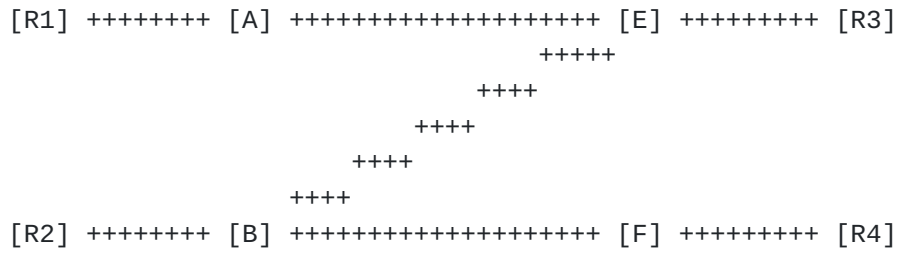

Figure 4c: Customized TE Topology merged with the Client's Native TE Topology

The data model proposed in this document can be used to retrieve/represent/manipulate the customized TE Topology depicted in Figure 4b.

4. Modeling Considerations

4.1. Generic network topology building blocks

The generic network topology building blocks are discussed in [YANG-NET-TOPO]. The TE Topology model proposed in this document augments and uses the ietf-network-topology module defined in [YANG-NET-TOPO].

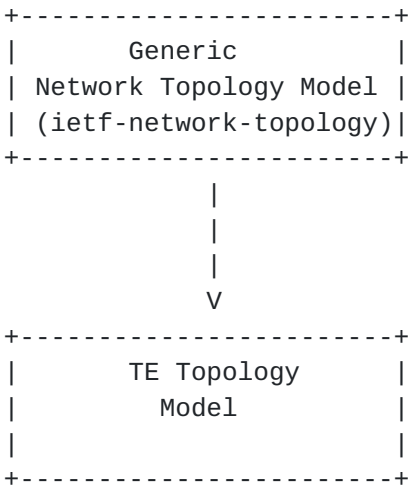

Figure 5: Augmenting the Generic Network Topology Model

4.2. Technology agnostic TE Topology model

The TE Topology model proposed in this document is meant to be technology agnostic. Other technology specific TE Topology models can augment and use the building blocks provided by the proposed model.

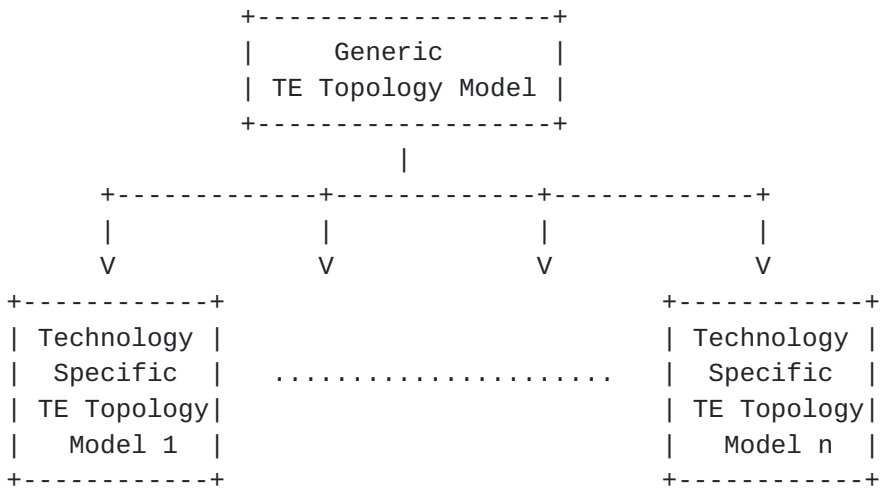

Figure 6: Augmenting the Technology agnostic TE Topology model

4.3. Model Structure

The high-level model structure proposed by this document is as shown below:

```
module: ietf-te-topology
augment /nw:network/nw:network-types:
   +--rw te-topology!
augment /nw:network:
   +--rw te!
      +--rw provider-id       te-global-id
      +--rw client-id         te-global-id
      +--rw te-topology-id    te-topology-id
      +--rw config
      |  .............
      +--ro state
      |  .............
      +--rw templates
         +--rw node-template* [name] {template}?
         |  .............
         +--rw link-template* [name] {template}?
            .............
```

```
augment /nw:network/nw:node:
   +--rw te!
      +--rw te-node-id    te-node-id
      +--rw config
      |  ............
      +--ro state
         ............

augment /nw:network/nt:link:
   +--rw te!
      +--rw config
      |  ..........
      +--ro state
         ..........

augment /nw:network/nw:node/nt:termination-point:
   +--rw te!
      +--rw te-tp-id    te-tp-id
      +--rw config
      |  ............
      +--ro state
         ............

notifications:
   +---n te-node-event
   |  ............
   +---n te-link-event
      ............
```

4.4. Topology Identifiers

The TE-Topology is uniquely identified by a key that has 3 constituents - te-topology-id, provider-id and client-id. The combination of provider-id and te-topology-id uniquely identifies a native TE Topology on a given provider. The client-id is used only when Customized TE Topologies come into play; a value of "0" is used as the client-id for native TE Topologies.

```
augment /nw:network:
   +--rw te!
      +--rw provider-id       te-global-id
      +--rw client-id         te-global-id
      +--rw te-topology-id    te-topology-id
```

4.5. Generic TE Link Attributes

The model covers the definitions for generic TE Link attributes - bandwidth, admin groups, SRLGs, switching capabilities, TE metric extensions etc.

```
  +--rw te-link-attributes
     .....................
     +--rw admin-status?                      te-admin-status
     +--rw performance-metric-throttle {te-performance-metric}?
     |  .....................
     +--rw link-index?                        uint64
     +--rw administrative-group?              te-types:admin-groups
     +--rw max-link-bandwidth?                decimal64
     +--rw max-resv-link-bandwidth?           decimal64
     +--rw unreserved-bandwidth* [priority]
     |  .....................
     +--rw te-default-metric?                 uint32
     +--rw performance-metric {te-performance-metric}?
     |  .....................
     +--rw link-protection-type?              enumeration
     +--rw interface-switching-capability* [switching-capability]
     |  .....................
     +--rw te-srlgs
        .....................
```

4.6. Generic TE Node Attributes

The model covers the definitions for generic TE Node attributes. The definition of a generic connectivity matrix is shown below:

```
  +--rw te-node-attributes
     ...........
     +--rw connectivity-matrix* [id]
     |  +--rw id            uint32
     |  +--rw from
     |  |  +--rw tp-ref?        leafref
     |  |  +--rw node-ref?      leafref
     |  |  +--rw network-ref?   leafref
     |  +--rw to
     |  |  +--rw tp-ref?        leafref
     |  |  +--rw node-ref?      leafref
     |  |  +--rw network-ref?   leafref
     |  +--rw is-allowed?   boolean
```

4.7. TED Information Sources

The model allows each TE topological element to have multiple TE information sources (OSPF-TE, ISIS-TE, BGP-LS, User-Configured, System-Processed, Other). Each information source is associated with a credibility preference to indicate precedence. In scenarios where a customized TE Topology is merged into a Client's native TE Topology, the merged topological elements would point to the corresponding customized TE Topology as its information source.

```
augment /nw:network/nw:node:
   +--rw te!
      ...........
      +--ro state
         ........
         +--ro information-source-state
            +--ro credibility-preference?   uint16
            +--ro topology
            |  +--ro provider-id-ref?      leafref
            |  +--ro client-id-ref?        leafref
            |  +--ro te-topology-id-ref?   leafref
            |  +--ro network-id-ref?       leafref
            +--ro routing-instance?         string

augment /nw:network/nt:link:
   +--rw te!
      ...........
      +--ro state
         .........
         +--ro information-source?         enumeration
         +--ro information-source-state
         |  +--ro credibility-preference?   uint16
         |  +--ro topology
         |  |  +--ro provider-id-ref?      leafref
         |  |  +--ro client-id-ref?        leafref
         |  |  +--ro te-topology-id-ref?   leafref
         |  |  +--ro network-id-ref?       leafref
         |  +--ro routing-instance?         string
         +--ro alt-information-sources* [information-source]
         |  ............
```

4.8. Overlay/Underlay Relationship

The model captures overlay and underlay relationship for TE nodes/links. For example - in networks where multiple TE Topologies

are built hierarchically, this model allows the user to start from a specific topological element in the top most topology and traverse all the way down to the supporting topological elements in the bottom most topology.

This relationship is captured via the "underlay-topology" field for the node and via the "underlay" field for the link. The use of these fields is optional and this functionality is tagged as a "feature" ("te-topology-hierarchy").

```
augment /nw:network/nw:node:
   +--rw te!
      +--rw te-node-id    te-node-id
      +--rw config
      |  +--rw te-node-template*     leafref {template}?
      |  +--rw te-node-attributes
      |     .....................
      |     +--rw underlay-topology {te-topology-hierarchy}?
      |        +--rw provider-id-ref?      leafref
      |        +--rw client-id-ref?        leafref
      |        +--rw te-topology-id-ref?   leafref
      |        +--rw network-id-ref?       leafref

augment /nw:network/nt:link:
   +--rw te!
      +--rw config
      |  .........
      |  +--rw te-link-attributes
      |     .....................
      |     +--rw underlay! {te-topology-hierarchy}?
      |     |  +--rw underlay-primary-path
      |     |  |  +--rw provider-id-ref?      leafref
      |     |  |  +--rw client-id-ref?        leafref
      |     |  |  +--rw te-topology-id-ref?   leafref
      |     |  |  +--rw network-id-ref?       leafref
      |     |  |  +--rw path-element* [path-element-id]
      |     |  |     ...............
      |     |  +--rw underlay-backup-path* [index]
      |     |  |  +--rw index                 uint32
      |     |  |  +--rw provider-id-ref?      leafref
      |     |  |  +--rw client-id-ref?        leafref
      |     |  |  +--rw te-topology-id-ref?   leafref
      |     |  |  +--rw network-id-ref?       leafref
      |     |  |  +--rw path-element* [path-element-id]
      |     |  |     ...............
      |     |  +--rw underlay-protection-type?   uint16
```

```
         |     |  +--rw underlay-trail-src
         |     |  |  ...........
         |     |  |  +--rw network-ref?   leafref
         |     |  +--rw underlay-trail-des
         |     |     ...........
```

4.9. Scheduling Parameters

The model allows time scheduling parameters to be specified for each topological element or for the topology as a whole. These parameters allow the provider to present different topological views to the client at different time slots. The use of "scheduling parameters" is optional and this functionality is tagged as a "feature" ("configuration-schedule"). [Editor's Note: The notion of "scheduling parameters" has wider applicability. The "schedules" module (which is imported by the TE Topology module) is discussed separately in Appendix A. The expectation is that this will eventually be discussed in a separate document.]

4.10. Templates

The data model provides the users with the ability to define templates and apply them to link and node configurations. The use of "template" configuration is optional and this functionality is tagged as a "feature" ("template").

```
  +--rw topology* [provider-id client-id te-topology-id]
  |  ...........
  |  +--rw node* [te-node-id]
  |  |  +--rw te-node-template?     leafref {template}?
  |  |     ..........
  |  +--rw link* [source-te-node-id source-te-link-id dest-te-node-
id dest-te-link-id]
  |     +--rw te-link-template?     leafref {template}?
  |        ..........
  |
  +--rw node-template* [name] {template}?
  |  +--rw name                        te-template-name
  |  +--rw priority?                   uint16
  |  +--rw reference-change-policy?    enumeration
  |  +--rw te-node-template*           leafref
  |  +--rw te-node-attributes
  |     ..........
  +--rw link-template* [name] {template}?
     +--rw name                        te-template-name
```

```
      +--rw priority?                  uint16
      +--rw reference-change-policy?   enumeration
      +--rw te-link-template*          leafref
      +--rw te-link-attributes
         ..........
```

A template can be constructed using multiple other templates. When two or more templates specify values for the same configuration field, the value from the template with the highest priority is used. The reference-change-policy specifies the action that needs to be taken when the template changes on a configuration node that has a reference to this template. The choices of action include taking no action, rejecting the change to the template and applying the change to the corresponding configuration. [Editor's Note: The notion of "templates" has wider applicability. It is possible for this to be discussed in a separate document.]

4.11. Notifications

Notifications are a key component of any topology data model.

[YANG-PUSH] defines a subscription and push mechanism for YANG datastores. This mechanism currently allows the user to:

- Subscribe notifications on a per client basis
- Specify subtree filters or xpath filters so that only interested contents will be sent.
- Specify either periodic or on-demand notifications.

The authors would like to recommend the use of this mechanism for the TE-Topology notifications. They would also like to suggest the following extensions to [YANG-PUSH]

- Specify specific entities that will trigger the push notifications. These entities can be specified by xpath, like the way a filter is specified.

- Specify or limit the triggering event type, e.g. "add", "delete", "modify", or "all". The system sends the push notifications only when such events happen on the triggering entities.

- Have an option to request either "incremental" or "full" notifications for an entity. For "incremental", the notification will contain only the changed attributes.

4.12. Open Items

- Coordinating changes to [YANG-PUSH]: The changes to [YANG-PUSH] discussed in Section 4.10 will need to be coordinated with the authors of that draft.

5. Tree Structure

```
module: ietf-te-topology
augment /nw:network/nw:network-types:
   +--rw te-topology!
augment /nw:network:
   +--rw te!
      +--rw provider-id       te-global-id
      +--rw client-id         te-global-id
      +--rw te-topology-id    te-topology-id
      +--rw config
      |  +--rw schedules
      |     +--rw schedule* [schedule-id]
      |        +--rw schedule-id          uint32
      |        +--rw start?               yang:date-and-time
      |        +--rw schedule-duration?   string
      |        +--rw repeat-interval?     string
      +--ro state
      |  +--ro schedules
      |     +--ro schedule* [schedule-id]
      |        +--ro schedule-id          uint32
      |        +--ro start?               yang:date-and-time
      |        +--ro schedule-duration?   string
      |        +--ro repeat-interval?     string
      +--rw templates
         +--rw node-template* [name] {template}?
         |  +--rw name                        te-template-name
         |  +--rw priority?                   uint16
         |  +--rw reference-change-policy?    enumeration
         |  +--rw te-node-attributes
         |     +--rw schedules
         |     |  +--rw schedule* [schedule-id]
         |     |     +--rw schedule-id          uint32
         |     |     +--rw start?               yang:date-and-time
         |     |     +--rw schedule-duration?   string
         |     |     +--rw repeat-interval?     string
         |     +--rw admin-status?          te-admin-status
         |     +--rw connectivity-matrix* [id]
         |     |  +--rw id            uint32
         |     |  +--rw from
```

```
            |     |  |  +--rw tp-ref?        leafref
            |     |  |  +--rw node-ref?      leafref
            |     |  |  +--rw network-ref?   leafref
            |     |  +--rw to
            |     |  |  +--rw tp-ref?        leafref
            |     |  |  +--rw node-ref?      leafref
            |     |  |  +--rw network-ref?   leafref
            |     |  +--rw is-allowed?   boolean
            |     +--rw domain-id?             uint32
            |     +--rw flag*                  te-node-flag
            |     +--rw is-abstract?           empty
            |     +--rw name?                  inet:domain-name
            |     +--rw signaling-address*     inet:ip-address
            |     +--rw underlay-topology {te-topology-hierarchy}?
            |        +--rw provider-id-ref?      leafref
            |        +--rw client-id-ref?        leafref
            |        +--rw te-topology-id-ref?   leafref
            |        +--rw network-id-ref?       leafref
            +--rw link-template* [name] {template}?
               +--rw name                       te-template-name
               +--rw priority?                  uint16
               +--rw reference-change-policy?   enumeration
               +--rw te-link-attributes
                  +--rw schedules
                  |  +--rw schedule* [schedule-id]
                  |     +--rw schedule-id          uint32
                  |     +--rw start?               yang:date-and-time
                  |     +--rw schedule-duration?   string
                  |     +--rw repeat-interval?     string
                  +--rw access-type?                       te-link-
   access-type
                  +--rw flag*                              te-link-flag
                  +--rw is-abstract?                       empty
                  +--rw name?                              string
                  +--rw underlay! {te-topology-hierarchy}?
                  |  +--rw underlay-primary-path
                  |  |  +--rw provider-id-ref?      leafref
                  |  |  +--rw client-id-ref?        leafref
                  |  |  +--rw te-topology-id-ref?   leafref
                  |  |  +--rw network-id-ref?       leafref
                  |  |  +--rw path-element* [path-element-id]
                  |  |     +--rw path-element-id     uint32
                  |  |     +--rw (type)?
                  |  |        +--:(ipv4-address)
                  |  |        |  +--rw v4-address?         inet:ipv4-
   address
```

```
                |  |        |  +--rw v4-prefix-length?   uint8
                |  |        |  +--rw v4-loose?           boolean
                |  |        +--:(ipv6-address)
                |  |        |  +--rw v6-address?         inet:ipv6-
   address
                |  |        |  +--rw v6-prefix-length?   uint8
                |  |        |  +--rw v6-loose?           boolean
                |  |        +--:(as-number)
                |  |        |  +--rw as-number?          uint16
                |  |        +--:(unnumbered-link)
                |  |        |  +--rw router-id?          inet:ip-
   address
                |  |        |  +--rw interface-id?       uint32
                |  |        +--:(label)
                |  |           +--rw value?             uint32
                |  +--rw underlay-backup-path* [index]
                |  |  +--rw index                 uint32
                |  |  +--rw provider-id-ref?      leafref
                |  |  +--rw client-id-ref?        leafref
                |  |  +--rw te-topology-id-ref?   leafref
                |  |  +--rw network-id-ref?       leafref
                |  |  +--rw path-element* [path-element-id]
                |  |     +--rw path-element-id     uint32
                |  |     +--rw (type)?
                |  |        +--:(ipv4-address)
                |  |        |  +--rw v4-address?         inet:ipv4-
   address
                |  |        |  +--rw v4-prefix-length?   uint8
                |  |        |  +--rw v4-loose?           boolean
                |  |        +--:(ipv6-address)
                |  |        |  +--rw v6-address?         inet:ipv6-
   address
                |  |        |  +--rw v6-prefix-length?   uint8
                |  |        |  +--rw v6-loose?           boolean
                |  |        +--:(as-number)
                |  |        |  +--rw as-number?          uint16
                |  |        +--:(unnumbered-link)
                |  |        |  +--rw router-id?          inet:ip-
   address
                |  |        |  +--rw interface-id?       uint32
                |  |        +--:(label)
                |  |           +--rw value?             uint32
                |  +--rw underlay-protection-type?   uint16
                |  +--rw underlay-trail-src
                |  |  +--rw tp-ref?        leafref
                |  |  +--rw node-ref?      leafref
```

```
                |  |  +--rw network-ref?   leafref
                |  +--rw underlay-trail-des
                |     +--rw tp-ref?        leafref
                |     +--rw node-ref?      leafref
                |     +--rw network-ref?   leafref
                +--rw admin-status?                    te-admin-
   status
                +--rw performance-metric-throttle {te-performance-
   metric}?
                |  +--rw unidirectional-delay-offset?
   uint32
                |  +--rw measure-interval?
   uint32
                |  +--rw advertisement-interval?
   uint32
                |  +--rw suppression-interval?
   uint32
                |  +--rw threshold-out
                |  |  +--rw unidirectional-delay?
   uint32
                |  |  +--rw unidirectional-min-delay?
   uint32
                |  |  +--rw unidirectional-max-delay?
   uint32
                |  |  +--rw unidirectional-delay-variation?
   uint32
                |  |  +--rw unidirectional-packet-loss?
   decimal64
                |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
                |  |  +--rw unidirectional-available-bandwidth?
   decimal64
                |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
                |  +--rw threshold-in
                |  |  +--rw unidirectional-delay?
   uint32
                |  |  +--rw unidirectional-min-delay?
   uint32
                |  |  +--rw unidirectional-max-delay?
   uint32
                |  |  +--rw unidirectional-delay-variation?
   uint32
                |  |  +--rw unidirectional-packet-loss?
   decimal64
```

```
                |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
                |  |  +--rw unidirectional-available-bandwidth?
   decimal64
                |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
                |  +--rw threshold-accelerated-advertisement
                |     +--rw unidirectional-delay?
   uint32
                |     +--rw unidirectional-min-delay?
   uint32
                |     +--rw unidirectional-max-delay?
   uint32
                |     +--rw unidirectional-delay-variation?
   uint32
                |     +--rw unidirectional-packet-loss?
   decimal64
                |     +--rw unidirectional-residual-bandwidth?
   decimal64
                |     +--rw unidirectional-available-bandwidth?
   decimal64
                |     +--rw unidirectional-utilized-bandwidth?
   decimal64
                +--rw link-index?                       uint64
                +--rw administrative-group?             te-
   types:admin-groups
                +--rw max-link-bandwidth?               decimal64
                +--rw max-resv-link-bandwidth?          decimal64
                +--rw unreserved-bandwidth* [priority]
                |  +--rw priority     uint8
                |  +--rw bandwidth?   decimal64
                +--rw te-default-metric?                uint32
                +--rw performance-metric {te-performance-metric}?
                |  +--rw measurement
                |  |  +--rw unidirectional-delay?
   uint32
                |  |  +--rw unidirectional-min-delay?
   uint32
                |  |  +--rw unidirectional-max-delay?
   uint32
                |  |  +--rw unidirectional-delay-variation?
   uint32
                |  |  +--rw unidirectional-packet-loss?
   decimal64
                |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
```

```
                |  |  +--rw unidirectional-available-bandwidth?
   decimal64
                |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
                |  +--rw normality
                |     +--rw unidirectional-delay?
   performance-metric-normality
                |     +--rw unidirectional-min-delay?
   performance-metric-normality
                |     +--rw unidirectional-max-delay?
   performance-metric-normality
                |     +--rw unidirectional-delay-variation?
   performance-metric-normality
                |     +--rw unidirectional-packet-loss?
   performance-metric-normality
                |     +--rw unidirectional-residual-bandwidth?
   performance-metric-normality
                |     +--rw unidirectional-available-bandwidth?
   performance-metric-normality
                |     +--rw unidirectional-utilized-bandwidth?
   performance-metric-normality
                +--rw link-protection-type?             enumeration
                +--rw interface-switching-capability* [switching-
   capability]
                |  +--rw switching-capability
   identityref
                |  +--rw encoding?
   identityref
                |  +--rw max-lsp-bandwidth* [priority]
                |  |  +--rw priority     uint8
                |  |  +--rw bandwidth?   decimal64
                |  +--rw packet-switch-capable
                |  |  +--rw minimum-lsp-bandwidth?   decimal64
                |  |  +--rw interface-mtu?           uint16
                |  +--rw time-division-multiplex-capable
                |  |  +--rw minimum-lsp-bandwidth?   decimal64
                |  |  +--rw indication?              enumeration
                |  +--rw interface-adjustment-capability* [upper-sc]
                |     +--rw upper-sc             identityref
                |     +--rw upper-encoding?      identityref
                |     +--rw max-lsp-bandwidth* [priority]
                |        +--rw priority     uint8
                |        +--rw bandwidth?   decimal64
                +--rw te-srlgs
                   +--rw values*   te-types:srlg
   augment /nw:network/nw:node:
```

```
      +--rw te!
         +--rw te-node-id    te-node-id
         +--rw config
         |  +--rw te-node-template*     leafref {template}?
         |  +--rw te-node-attributes
         |     +--rw schedules
         |     |  +--rw schedule* [schedule-id]
         |     |     +--rw schedule-id          uint32
         |     |     +--rw start?               yang:date-and-time
         |     |     +--rw schedule-duration?   string
         |     |     +--rw repeat-interval?     string
         |     +--rw admin-status?          te-admin-status
         |     +--rw connectivity-matrix* [id]
         |     |  +--rw id            uint32
         |     |  +--rw from
         |     |  |  +--rw tp-ref?        leafref
         |     |  |  +--rw node-ref?      leafref
         |     |  |  +--rw network-ref?   leafref
         |     |  +--rw to
         |     |  |  +--rw tp-ref?        leafref
         |     |  |  +--rw node-ref?      leafref
         |     |  |  +--rw network-ref?   leafref
         |     |  +--rw is-allowed?   boolean
         |     +--rw domain-id?             uint32
         |     +--rw flag*                  te-node-flag
         |     +--rw is-abstract?           empty
         |     +--rw name?                  inet:domain-name
         |     +--rw signaling-address*     inet:ip-address
         |     +--rw underlay-topology {te-topology-hierarchy}?
         |        +--rw provider-id-ref?      leafref
         |        +--rw client-id-ref?        leafref
         |        +--rw te-topology-id-ref?   leafref
         |        +--rw network-id-ref?       leafref
         +--ro state
            +--ro te-node-template*           leafref {template}?
            +--ro te-node-attributes
            |  +--ro schedules
            |  |  +--ro schedule* [schedule-id]
            |  |     +--ro schedule-id          uint32
            |  |     +--ro start?               yang:date-and-time
            |  |     +--ro schedule-duration?   string
            |  |     +--ro repeat-interval?     string
            |  +--ro admin-status?          te-admin-status
            |  +--ro connectivity-matrix* [id]
            |  |  +--ro id            uint32
            |  |  +--ro from
```

```
            |  |  |  +--ro tp-ref?        leafref
            |  |  |  +--ro node-ref?      leafref
            |  |  |  +--ro network-ref?   leafref
            |  |  +--ro to
            |  |  |  +--ro tp-ref?        leafref
            |  |  |  +--ro node-ref?      leafref
            |  |  |  +--ro network-ref?   leafref
            |  |  +--ro is-allowed?   boolean
            |  +--ro domain-id?             uint32
            |  +--ro flag*                  te-node-flag
            |  +--ro is-abstract?           empty
            |  +--ro name?                  inet:domain-name
            |  +--ro signaling-address*     inet:ip-address
            |  +--ro underlay-topology {te-topology-hierarchy}?
            |     +--ro provider-id-ref?      leafref
            |     +--ro client-id-ref?        leafref
            |     +--ro te-topology-id-ref?   leafref
            |     +--ro network-id-ref?       leafref
            +--ro oper-status?                te-oper-status
            +--ro is-multi-access-dr?         empty
            +--ro information-source?         enumeration
            +--ro information-source-state
               +--ro credibility-preference?   uint16
               +--ro topology
               |  +--ro provider-id-ref?      leafref
               |  +--ro client-id-ref?        leafref
               |  +--ro te-topology-id-ref?   leafref
               |  +--ro network-id-ref?       leafref
               +--ro routing-instance?         string
   augment /nw:network/nt:link:
      +--rw te!
         +--rw config
         |  +--rw (bundle-stack-level)?
         |  |  +--:(bundle)
         |  |  |  +--rw bundled-links
         |  |  |     +--rw bundled-link* [sequence]
         |  |  |        +--rw sequence      uint32
         |  |  |        +--rw src-tp-ref?   leafref
         |  |  |        +--rw des-tp-ref?   leafref
         |  |  +--:(component)
         |  |     +--rw component-links
         |  |        +--rw component-link* [sequence]
         |  |           +--rw sequence             uint32
         |  |           +--rw src-interface-ref?   string
         |  |           +--rw des-interface-ref?   string
         |  +--rw te-link-template*     leafref {template}?
```

```
         |  +--rw te-link-attributes
         |     +--rw schedules
         |     |  +--rw schedule* [schedule-id]
         |     |     +--rw schedule-id          uint32
         |     |     +--rw start?               yang:date-and-time
         |     |     +--rw schedule-duration?   string
         |     |     +--rw repeat-interval?     string
         |     +--rw access-type?                       te-link-access-
   type
         |     +--rw flag*                              te-link-flag
         |     +--rw is-abstract?                       empty
         |     +--rw name?                              string
         |     +--rw underlay! {te-topology-hierarchy}?
         |     |  +--rw underlay-primary-path
         |     |  |  +--rw provider-id-ref?      leafref
         |     |  |  +--rw client-id-ref?        leafref
         |     |  |  +--rw te-topology-id-ref?   leafref
         |     |  |  +--rw network-id-ref?       leafref
         |     |  |  +--rw path-element* [path-element-id]
         |     |  |     +--rw path-element-id     uint32
         |     |  |     +--rw (type)?
         |     |  |        +--:(ipv4-address)
         |     |  |        |  +--rw v4-address?         inet:ipv4-
   address
         |     |  |        |  +--rw v4-prefix-length?   uint8
         |     |  |        |  +--rw v4-loose?           boolean
         |     |  |        +--:(ipv6-address)
         |     |  |        |  +--rw v6-address?         inet:ipv6-
   address
         |     |  |        |  +--rw v6-prefix-length?   uint8
         |     |  |        |  +--rw v6-loose?           boolean
         |     |  |        +--:(as-number)
         |     |  |        |  +--rw as-number?          uint16
         |     |  |        +--:(unnumbered-link)
         |     |  |        |  +--rw router-id?          inet:ip-address
         |     |  |        |  +--rw interface-id?       uint32
         |     |  |        +--:(label)
         |     |  |           +--rw value?             uint32
         |     |  +--rw underlay-backup-path* [index]
         |     |  |  +--rw index                 uint32
         |     |  |  +--rw provider-id-ref?      leafref
         |     |  |  +--rw client-id-ref?        leafref
         |     |  |  +--rw te-topology-id-ref?   leafref
         |     |  |  +--rw network-id-ref?       leafref
         |     |  |  +--rw path-element* [path-element-id]
         |     |  |     +--rw path-element-id     uint32
```

```
        |     |  |     +--rw (type)?
        |     |  |        +--:(ipv4-address)
        |     |  |        |  +--rw v4-address?         inet:ipv4-
   address
        |     |  |        |  +--rw v4-prefix-length?   uint8
        |     |  |        |  +--rw v4-loose?           boolean
        |     |  |        +--:(ipv6-address)
        |     |  |        |  +--rw v6-address?         inet:ipv6-
   address
        |     |  |        |  +--rw v6-prefix-length?   uint8
        |     |  |        |  +--rw v6-loose?           boolean
        |     |  |        +--:(as-number)
        |     |  |        |  +--rw as-number?          uint16
        |     |  |        +--:(unnumbered-link)
        |     |  |        |  +--rw router-id?          inet:ip-address
        |     |  |        |  +--rw interface-id?       uint32
        |     |  |        +--:(label)
        |     |  |           +--rw value?             uint32
        |     |  +--rw underlay-protection-type?   uint16
        |     |  +--rw underlay-trail-src
        |     |  |  +--rw tp-ref?        leafref
        |     |  |  +--rw node-ref?      leafref
        |     |  |  +--rw network-ref?   leafref
        |     |  +--rw underlay-trail-des
        |     |     +--rw tp-ref?        leafref
        |     |     +--rw node-ref?      leafref
        |     |     +--rw network-ref?   leafref
        |     +--rw admin-status?                    te-admin-status
        |     +--rw performance-metric-throttle {te-performance-
   metric}?
        |     |  +--rw unidirectional-delay-offset?          uint32
        |     |  +--rw measure-interval?                     uint32
        |     |  +--rw advertisement-interval?               uint32
        |     |  +--rw suppression-interval?                 uint32
        |     |  +--rw threshold-out
        |     |  |  +--rw unidirectional-delay?                uint32
        |     |  |  +--rw unidirectional-min-delay?            uint32
        |     |  |  +--rw unidirectional-max-delay?            uint32
        |     |  |  +--rw unidirectional-delay-variation?      uint32
        |     |  |  +--rw unidirectional-packet-loss?
   decimal64
        |     |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
        |     |  |  +--rw unidirectional-available-bandwidth?
   decimal64
```

```
        |     |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
        |     |  +--rw threshold-in
        |     |  |  +--rw unidirectional-delay?                uint32
        |     |  |  +--rw unidirectional-min-delay?            uint32
        |     |  |  +--rw unidirectional-max-delay?            uint32
        |     |  |  +--rw unidirectional-delay-variation?      uint32
        |     |  |  +--rw unidirectional-packet-loss?
   decimal64
        |     |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
        |     |  |  +--rw unidirectional-available-bandwidth?
   decimal64
        |     |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
        |     |  +--rw threshold-accelerated-advertisement
        |     |     +--rw unidirectional-delay?                uint32
        |     |     +--rw unidirectional-min-delay?            uint32
        |     |     +--rw unidirectional-max-delay?            uint32
        |     |     +--rw unidirectional-delay-variation?      uint32
        |     |     +--rw unidirectional-packet-loss?
   decimal64
        |     |     +--rw unidirectional-residual-bandwidth?
   decimal64
        |     |     +--rw unidirectional-available-bandwidth?
   decimal64
        |     |     +--rw unidirectional-utilized-bandwidth?
   decimal64
        |     +--rw link-index?                       uint64
        |     +--rw administrative-group?             te-types:admin-
   groups
        |     +--rw max-link-bandwidth?               decimal64
        |     +--rw max-resv-link-bandwidth?          decimal64
        |     +--rw unreserved-bandwidth* [priority]
        |     |  +--rw priority     uint8
        |     |  +--rw bandwidth?   decimal64
        |     +--rw te-default-metric?                uint32
        |     +--rw performance-metric {te-performance-metric}?
        |     |  +--rw measurement
        |     |  |  +--rw unidirectional-delay?                uint32
        |     |  |  +--rw unidirectional-min-delay?            uint32
        |     |  |  +--rw unidirectional-max-delay?            uint32
        |     |  |  +--rw unidirectional-delay-variation?      uint32
        |     |  |  +--rw unidirectional-packet-loss?
   decimal64
```

```
      |     |  |  +--rw unidirectional-residual-bandwidth?
   decimal64
      |     |  |  +--rw unidirectional-available-bandwidth?
   decimal64
      |     |  |  +--rw unidirectional-utilized-bandwidth?
   decimal64
      |     |  +--rw normality
      |     |     +--rw unidirectional-delay?
   performance-metric-normality
      |     |     +--rw unidirectional-min-delay?
   performance-metric-normality
      |     |     +--rw unidirectional-max-delay?
   performance-metric-normality
      |     |     +--rw unidirectional-delay-variation?
   performance-metric-normality
      |     |     +--rw unidirectional-packet-loss?
   performance-metric-normality
      |     |     +--rw unidirectional-residual-bandwidth?
   performance-metric-normality
      |     |     +--rw unidirectional-available-bandwidth?
   performance-metric-normality
      |     |     +--rw unidirectional-utilized-bandwidth?
   performance-metric-normality
      |     +--rw link-protection-type?             enumeration
      |     +--rw interface-switching-capability* [switching-
   capability]
      |     |  +--rw switching-capability               identityref
      |     |  +--rw encoding?                          identityref
      |     |  +--rw max-lsp-bandwidth* [priority]
      |     |  |  +--rw priority     uint8
      |     |  |  +--rw bandwidth?   decimal64
      |     |  +--rw packet-switch-capable
      |     |  |  +--rw minimum-lsp-bandwidth?   decimal64
      |     |  |  +--rw interface-mtu?           uint16
      |     |  +--rw time-division-multiplex-capable
      |     |  |  +--rw minimum-lsp-bandwidth?   decimal64
      |     |  |  +--rw indication?              enumeration
      |     |  +--rw interface-adjustment-capability* [upper-sc]
      |     |     +--rw upper-sc             identityref
      |     |     +--rw upper-encoding?      identityref
      |     |     +--rw max-lsp-bandwidth* [priority]
      |     |        +--rw priority     uint8
      |     |        +--rw bandwidth?   decimal64
      |     +--rw te-srlgs
      |        +--rw values*   te-types:srlg
      +--ro state
```

```
            +--ro (bundle-stack-level)?
            |  +--:(bundle)
            |  |  +--ro bundled-links
            |  |     +--ro bundled-link* [sequence]
            |  |        +--ro sequence      uint32
            |  |        +--ro src-tp-ref?   leafref
            |  |        +--ro des-tp-ref?   leafref
            |  +--:(component)
            |     +--ro component-links
            |        +--ro component-link* [sequence]
            |           +--ro sequence             uint32
            |           +--ro src-interface-ref?   string
            |           +--ro des-interface-ref?   string
            +--ro te-link-template*           leafref {template}?
            +--ro te-link-attributes
            |  +--ro schedules
            |  |  +--ro schedule* [schedule-id]
            |  |     +--ro schedule-id          uint32
            |  |     +--ro start?               yang:date-and-time
            |  |     +--ro schedule-duration?   string
            |  |     +--ro repeat-interval?     string
            |  +--ro access-type?                      te-link-access-
   type
            |  +--ro flag*                             te-link-flag
            |  +--ro is-abstract?                      empty
            |  +--ro name?                             string
            |  +--ro underlay! {te-topology-hierarchy}?
            |  |  +--ro underlay-primary-path
            |  |  |  +--ro provider-id-ref?      leafref
            |  |  |  +--ro client-id-ref?        leafref
            |  |  |  +--ro te-topology-id-ref?   leafref
            |  |  |  +--ro network-id-ref?       leafref
            |  |  |  +--ro path-element* [path-element-id]
            |  |  |     +--ro path-element-id     uint32
            |  |  |     +--ro (type)?
            |  |  |        +--:(ipv4-address)
            |  |  |        |  +--ro v4-address?         inet:ipv4-
   address
            |  |  |        |  +--ro v4-prefix-length?   uint8
            |  |  |        |  +--ro v4-loose?           boolean
            |  |  |        +--:(ipv6-address)
            |  |  |        |  +--ro v6-address?         inet:ipv6-
   address
            |  |  |        |  +--ro v6-prefix-length?   uint8
            |  |  |        |  +--ro v6-loose?           boolean
            |  |  |        +--:(as-number)
```

```
          |  |  |        |  +--ro as-number?          uint16
          |  |  |        +--:(unnumbered-link)
          |  |  |        |  +--ro router-id?          inet:ip-address
          |  |  |        |  +--ro interface-id?       uint32
          |  |  |        +--:(label)
          |  |  |           +--ro value?              uint32
          |  |  +--ro underlay-backup-path* [index]
          |  |  |  +--ro index                 uint32
          |  |  |  +--ro provider-id-ref?      leafref
          |  |  |  +--ro client-id-ref?        leafref
          |  |  |  +--ro te-topology-id-ref?   leafref
          |  |  |  +--ro network-id-ref?       leafref
          |  |  |  +--ro path-element* [path-element-id]
          |  |  |     +--ro path-element-id     uint32
          |  |  |     +--ro (type)?
          |  |  |        +--:(ipv4-address)
          |  |  |        |  +--ro v4-address?         inet:ipv4-
   address
          |  |  |        |  +--ro v4-prefix-length?   uint8
          |  |  |        |  +--ro v4-loose?           boolean
          |  |  |        +--:(ipv6-address)
          |  |  |        |  +--ro v6-address?         inet:ipv6-
   address
          |  |  |        |  +--ro v6-prefix-length?   uint8
          |  |  |        |  +--ro v6-loose?           boolean
          |  |  |        +--:(as-number)
          |  |  |        |  +--ro as-number?          uint16
          |  |  |        +--:(unnumbered-link)
          |  |  |        |  +--ro router-id?          inet:ip-address
          |  |  |        |  +--ro interface-id?       uint32
          |  |  |        +--:(label)
          |  |  |           +--ro value?              uint32
          |  |  +--ro underlay-protection-type?   uint16
          |  |  +--ro underlay-trail-src
          |  |  |  +--ro tp-ref?        leafref
          |  |  |  +--ro node-ref?      leafref
          |  |  |  +--ro network-ref?   leafref
          |  |  +--ro underlay-trail-des
          |  |     +--ro tp-ref?        leafref
          |  |     +--ro node-ref?      leafref
          |  |     +--ro network-ref?   leafref
          |  +--ro admin-status?                    te-admin-status
          |  +--ro performance-metric-throttle {te-performance-
   metric}?
          |  |  +--ro unidirectional-delay-offset?          uint32
          |  |  +--ro measure-interval?                     uint32
```

```
          |  |  +--ro advertisement-interval?               uint32
          |  |  +--ro suppression-interval?                 uint32
          |  |  +--ro threshold-out
          |  |  |  +--ro unidirectional-delay?                uint32
          |  |  |  +--ro unidirectional-min-delay?            uint32
          |  |  |  +--ro unidirectional-max-delay?            uint32
          |  |  |  +--ro unidirectional-delay-variation?      uint32
          |  |  |  +--ro unidirectional-packet-loss?
   decimal64
          |  |  |  +--ro unidirectional-residual-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-available-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-utilized-bandwidth?
   decimal64
          |  |  +--ro threshold-in
          |  |  |  +--ro unidirectional-delay?                uint32
          |  |  |  +--ro unidirectional-min-delay?            uint32
          |  |  |  +--ro unidirectional-max-delay?            uint32
          |  |  |  +--ro unidirectional-delay-variation?      uint32
          |  |  |  +--ro unidirectional-packet-loss?
   decimal64
          |  |  |  +--ro unidirectional-residual-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-available-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-utilized-bandwidth?
   decimal64
          |  |  +--ro threshold-accelerated-advertisement
          |  |     +--ro unidirectional-delay?                uint32
          |  |     +--ro unidirectional-min-delay?            uint32
          |  |     +--ro unidirectional-max-delay?            uint32
          |  |     +--ro unidirectional-delay-variation?      uint32
          |  |     +--ro unidirectional-packet-loss?
   decimal64
          |  |     +--ro unidirectional-residual-bandwidth?
   decimal64
          |  |     +--ro unidirectional-available-bandwidth?
   decimal64
          |  |     +--ro unidirectional-utilized-bandwidth?
   decimal64
          |  +--ro link-index?                        uint64
          |  +--ro administrative-group?              te-types:admin-
   groups
          |  +--ro max-link-bandwidth?                decimal64
          |  +--ro max-resv-link-bandwidth?           decimal64
```

```
          |  +--ro unreserved-bandwidth* [priority]
          |  |  +--ro priority     uint8
          |  |  +--ro bandwidth?   decimal64
          |  +--ro te-default-metric?               uint32
          |  +--ro performance-metric {te-performance-metric}?
          |  |  +--ro measurement
          |  |  |  +--ro unidirectional-delay?               uint32
          |  |  |  +--ro unidirectional-min-delay?           uint32
          |  |  |  +--ro unidirectional-max-delay?           uint32
          |  |  |  +--ro unidirectional-delay-variation?     uint32
          |  |  |  +--ro unidirectional-packet-loss?
   decimal64
          |  |  |  +--ro unidirectional-residual-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-available-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-utilized-bandwidth?
   decimal64
          |  |  +--ro normality
          |  |     +--ro unidirectional-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-min-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-max-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-delay-variation?
   performance-metric-normality
          |  |     +--ro unidirectional-packet-loss?
   performance-metric-normality
          |  |     +--ro unidirectional-residual-bandwidth?
   performance-metric-normality
          |  |     +--ro unidirectional-available-bandwidth?
   performance-metric-normality
          |  |     +--ro unidirectional-utilized-bandwidth?
   performance-metric-normality
          |  +--ro link-protection-type?             enumeration
          |  +--ro interface-switching-capability* [switching-
   capability]
          |  |  +--ro switching-capability              identityref
          |  |  +--ro encoding?                         identityref
          |  |  +--ro max-lsp-bandwidth* [priority]
          |  |  |  +--ro priority     uint8
          |  |  |  +--ro bandwidth?   decimal64
          |  |  +--ro packet-switch-capable
          |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
          |  |  |  +--ro interface-mtu?           uint16
```

```
            |  |  +--ro time-division-multiplex-capable
            |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
            |  |  |  +--ro indication?              enumeration
            |  |  +--ro interface-adjustment-capability* [upper-sc]
            |  |     +--ro upper-sc             identityref
            |  |     +--ro upper-encoding?      identityref
            |  |     +--ro max-lsp-bandwidth* [priority]
            |  |        +--ro priority     uint8
            |  |        +--ro bandwidth?   decimal64
            |  +--ro te-srlgs
            |     +--ro values*   te-types:srlg
            +--ro oper-status?               te-oper-status
            +--ro information-source?        enumeration
            +--ro information-source-state
            |  +--ro credibility-preference?   uint16
            |  +--ro topology
            |  |  +--ro provider-id-ref?      leafref
            |  |  +--ro client-id-ref?        leafref
            |  |  +--ro te-topology-id-ref?   leafref
            |  |  +--ro network-id-ref?       leafref
            |  +--ro routing-instance?         string
            +--ro alt-information-sources* [information-source]
            |  +--ro information-source               enumeration
            |  +--ro information-source-state
            |  |  +--ro credibility-preference?   uint16
            |  |  +--ro topology
            |  |  |  +--ro provider-id-ref?      leafref
            |  |  |  +--ro client-id-ref?        leafref
            |  |  |  +--ro te-topology-id-ref?   leafref
            |  |  |  +--ro network-id-ref?       leafref
            |  |  +--ro routing-instance?         string
            |  +--ro link-index?                      uint64
            |  +--ro administrative-group?            te-types:admin-
   groups
            |  +--ro max-link-bandwidth?              decimal64
            |  +--ro max-resv-link-bandwidth?         decimal64
            |  +--ro unreserved-bandwidth* [priority]
            |  |  +--ro priority     uint8
            |  |  +--ro bandwidth?   decimal64
            |  +--ro te-default-metric?               uint32
            |  +--ro performance-metric {te-performance-metric}?
            |  |  +--ro measurement
            |  |  |  +--ro unidirectional-delay?                uint32
            |  |  |  +--ro unidirectional-min-delay?            uint32
            |  |  |  +--ro unidirectional-max-delay?            uint32
            |  |  |  +--ro unidirectional-delay-variation?      uint32
```

```
          |  |  |  +--ro unidirectional-packet-loss?
   decimal64
          |  |  |  +--ro unidirectional-residual-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-available-bandwidth?
   decimal64
          |  |  |  +--ro unidirectional-utilized-bandwidth?
   decimal64
          |  |  +--ro normality
          |  |     +--ro unidirectional-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-min-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-max-delay?
   performance-metric-normality
          |  |     +--ro unidirectional-delay-variation?
   performance-metric-normality
          |  |     +--ro unidirectional-packet-loss?
   performance-metric-normality
          |  |     +--ro unidirectional-residual-bandwidth?
   performance-metric-normality
          |  |     +--ro unidirectional-available-bandwidth?
   performance-metric-normality
          |  |     +--ro unidirectional-utilized-bandwidth?
   performance-metric-normality
          |  +--ro link-protection-type?             enumeration
          |  +--ro interface-switching-capability* [switching-
   capability]
          |  |  +--ro switching-capability              identityref
          |  |  +--ro encoding?                         identityref
          |  |  +--ro max-lsp-bandwidth* [priority]
          |  |  |  +--ro priority     uint8
          |  |  |  +--ro bandwidth?   decimal64
          |  |  +--ro packet-switch-capable
          |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
          |  |  |  +--ro interface-mtu?           uint16
          |  |  +--ro time-division-multiplex-capable
          |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
          |  |  |  +--ro indication?              enumeration
          |  |  +--ro interface-adjustment-capability* [upper-sc]
          |  |     +--ro upper-sc             identityref
          |  |     +--ro upper-encoding?      identityref
          |  |     +--ro max-lsp-bandwidth* [priority]
          |  |        +--ro priority     uint8
          |  |        +--ro bandwidth?   decimal64
          |  +--ro te-srlgs
```

```
            |     +--ro values*   te-types:srlg
            +--ro recovery
            |  +--ro restoration-status?   te-recovery-status
            |  +--ro protection-status?    te-recovery-status
            +--ro underlay {te-topology-hierarchy}?
               +--ro dynamic?     boolean
               +--ro committed?   boolean
   augment /nw:network/nw:node/nt:termination-point:
      +--rw te!
         +--rw te-tp-id    te-tp-id
         +--rw config
         |  +--rw schedules
         |     +--rw schedule* [schedule-id]
         |        +--rw schedule-id          uint32
         |        +--rw start?               yang:date-and-time
         |        +--rw schedule-duration?   string
         |        +--rw repeat-interval?     string
         +--ro state
            +--ro schedules
               +--ro schedule* [schedule-id]
                  +--ro schedule-id          uint32
                  +--ro start?               yang:date-and-time
                  +--ro schedule-duration?   string
                  +--ro repeat-interval?     string
   notifications:
      +---n te-node-event
      |  +--ro event-type?                te-topology-event-type
      |  +--ro node-ref?                  leafref
      |  +--ro network-ref?               leafref
      |  +--ro te-topology!
      |  +--ro te-node-attributes
      |  |  +--ro schedules
      |  |  |  +--ro schedule* [schedule-id]
      |  |  |     +--ro schedule-id          uint32
      |  |  |     +--ro start?               yang:date-and-time
      |  |  |     +--ro schedule-duration?   string
      |  |  |     +--ro repeat-interval?     string
      |  |  +--ro admin-status?          te-admin-status
      |  |  +--ro connectivity-matrix* [id]
      |  |  |  +--ro id            uint32
      |  |  |  +--ro from
      |  |  |  |  +--ro tp-ref?        leafref
      |  |  |  |  +--ro node-ref?      leafref
      |  |  |  |  +--ro network-ref?   leafref
      |  |  |  +--ro to
      |  |  |  |  +--ro tp-ref?        leafref
```

```
      |  |  |  |  +--ro node-ref?      leafref
      |  |  |  |  +--ro network-ref?   leafref
      |  |  |  +--ro is-allowed?   boolean
      |  |  +--ro domain-id?             uint32
      |  |  +--ro flag*                  te-node-flag
      |  |  +--ro is-abstract?           empty
      |  |  +--ro name?                  inet:domain-name
      |  |  +--ro signaling-address*     inet:ip-address
      |  |  +--ro underlay-topology {te-topology-hierarchy}?
      |  |     +--ro provider-id-ref?      leafref
      |  |     +--ro client-id-ref?        leafref
      |  |     +--ro te-topology-id-ref?   leafref
      |  |     +--ro network-id-ref?       leafref
      |  +--ro oper-status?               te-oper-status
      |  +--ro is-multi-access-dr?        empty
      |  +--ro information-source?        enumeration
      |  +--ro information-source-state
      |     +--ro credibility-preference?   uint16
      |     +--ro topology
      |     |  +--ro provider-id-ref?      leafref
      |     |  +--ro client-id-ref?        leafref
      |     |  +--ro te-topology-id-ref?   leafref
      |     |  +--ro network-id-ref?       leafref
      |     +--ro routing-instance?         string
      +---n te-link-event
         +--ro event-type?                te-topology-event-type
         +--ro link-ref?                  leafref
         +--ro network-ref?               leafref
         +--ro te-topology!
         +--ro te-link-attributes
         |  +--ro schedules
         |  |  +--ro schedule* [schedule-id]
         |  |     +--ro schedule-id          uint32
         |  |     +--ro start?               yang:date-and-time
         |  |     +--ro schedule-duration?   string
         |  |     +--ro repeat-interval?     string
         |  +--ro access-type?                    te-link-access-type
         |  +--ro flag*                           te-link-flag
         |  +--ro is-abstract?                    empty
         |  +--ro name?                           string
         |  +--ro underlay! {te-topology-hierarchy}?
         |  |  +--ro underlay-primary-path
         |  |  |  +--ro provider-id-ref?      leafref
         |  |  |  +--ro client-id-ref?        leafref
         |  |  |  +--ro te-topology-id-ref?   leafref
         |  |  |  +--ro network-id-ref?       leafref
```

```
          |  |  |  +--ro path-element* [path-element-id]
          |  |  |     +--ro path-element-id     uint32
          |  |  |     +--ro (type)?
          |  |  |        +--:(ipv4-address)
          |  |  |        |  +--ro v4-address?         inet:ipv4-address
          |  |  |        |  +--ro v4-prefix-length?   uint8
          |  |  |        |  +--ro v4-loose?           boolean
          |  |  |        +--:(ipv6-address)
          |  |  |        |  +--ro v6-address?         inet:ipv6-address
          |  |  |        |  +--ro v6-prefix-length?   uint8
          |  |  |        |  +--ro v6-loose?           boolean
          |  |  |        +--:(as-number)
          |  |  |        |  +--ro as-number?          uint16
          |  |  |        +--:(unnumbered-link)
          |  |  |        |  +--ro router-id?          inet:ip-address
          |  |  |        |  +--ro interface-id?       uint32
          |  |  |        +--:(label)
          |  |  |           +--ro value?             uint32
          |  |  +--ro underlay-backup-path* [index]
          |  |  |  +--ro index                 uint32
          |  |  |  +--ro provider-id-ref?      leafref
          |  |  |  +--ro client-id-ref?        leafref
          |  |  |  +--ro te-topology-id-ref?   leafref
          |  |  |  +--ro network-id-ref?       leafref
          |  |  |  +--ro path-element* [path-element-id]
          |  |  |     +--ro path-element-id     uint32
          |  |  |     +--ro (type)?
          |  |  |        +--:(ipv4-address)
          |  |  |        |  +--ro v4-address?         inet:ipv4-address
          |  |  |        |  +--ro v4-prefix-length?   uint8
          |  |  |        |  +--ro v4-loose?           boolean
          |  |  |        +--:(ipv6-address)
          |  |  |        |  +--ro v6-address?         inet:ipv6-address
          |  |  |        |  +--ro v6-prefix-length?   uint8
          |  |  |        |  +--ro v6-loose?           boolean
          |  |  |        +--:(as-number)
          |  |  |        |  +--ro as-number?          uint16
          |  |  |        +--:(unnumbered-link)
          |  |  |        |  +--ro router-id?          inet:ip-address
          |  |  |        |  +--ro interface-id?       uint32
          |  |  |        +--:(label)
          |  |  |           +--ro value?             uint32
          |  |  +--ro underlay-protection-type?   uint16
          |  |  +--ro underlay-trail-src
          |  |  |  +--ro tp-ref?        leafref
          |  |  |  +--ro node-ref?      leafref
```

```
         |  |  |  +--ro network-ref?   leafref
         |  |  +--ro underlay-trail-des
         |  |  |  +--ro tp-ref?        leafref
         |  |  |  +--ro node-ref?      leafref
         |  |  |  +--ro network-ref?   leafref
         |  |  +--ro dynamic?                    boolean
         |  |  +--ro committed?                  boolean
         |  +--ro admin-status?                     te-admin-status
         |  +--ro performance-metric-throttle {te-performance-metric}?
         |  |  +--ro unidirectional-delay-offset?           uint32
         |  |  +--ro measure-interval?                      uint32
         |  |  +--ro advertisement-interval?                uint32
         |  |  +--ro suppression-interval?                  uint32
         |  |  +--ro threshold-out
         |  |  |  +--ro unidirectional-delay?                 uint32
         |  |  |  +--ro unidirectional-min-delay?             uint32
         |  |  |  +--ro unidirectional-max-delay?             uint32
         |  |  |  +--ro unidirectional-delay-variation?       uint32
         |  |  |  +--ro unidirectional-packet-loss?           decimal64
         |  |  |  +--ro unidirectional-residual-bandwidth?    decimal64
         |  |  |  +--ro unidirectional-available-bandwidth?   decimal64
         |  |  |  +--ro unidirectional-utilized-bandwidth?    decimal64
         |  |  +--ro threshold-in
         |  |  |  +--ro unidirectional-delay?                 uint32
         |  |  |  +--ro unidirectional-min-delay?             uint32
         |  |  |  +--ro unidirectional-max-delay?             uint32
         |  |  |  +--ro unidirectional-delay-variation?       uint32
         |  |  |  +--ro unidirectional-packet-loss?           decimal64
         |  |  |  +--ro unidirectional-residual-bandwidth?    decimal64
         |  |  |  +--ro unidirectional-available-bandwidth?   decimal64
         |  |  |  +--ro unidirectional-utilized-bandwidth?    decimal64
         |  |  +--ro threshold-accelerated-advertisement
         |  |     +--ro unidirectional-delay?                 uint32
         |  |     +--ro unidirectional-min-delay?             uint32
         |  |     +--ro unidirectional-max-delay?             uint32
         |  |     +--ro unidirectional-delay-variation?       uint32
         |  |     +--ro unidirectional-packet-loss?           decimal64
         |  |     +--ro unidirectional-residual-bandwidth?    decimal64
         |  |     +--ro unidirectional-available-bandwidth?   decimal64
         |  |     +--ro unidirectional-utilized-bandwidth?    decimal64
         |  +--ro link-index?                        uint64
         |  +--ro administrative-group?              te-types:admin-
   groups
         |  +--ro max-link-bandwidth?                decimal64
         |  +--ro max-resv-link-bandwidth?           decimal64
         |  +--ro unreserved-bandwidth* [priority]
```

```
         |  |  +--ro priority     uint8
         |  |  +--ro bandwidth?   decimal64
         |  +--ro te-default-metric?                uint32
         |  +--ro performance-metric {te-performance-metric}?
         |  |  +--ro measurement
         |  |  |  +--ro unidirectional-delay?                 uint32
         |  |  |  +--ro unidirectional-min-delay?             uint32
         |  |  |  +--ro unidirectional-max-delay?             uint32
         |  |  |  +--ro unidirectional-delay-variation?       uint32
         |  |  |  +--ro unidirectional-packet-loss?           decimal64
         |  |  |  +--ro unidirectional-residual-bandwidth?    decimal64
         |  |  |  +--ro unidirectional-available-bandwidth?   decimal64
         |  |  |  +--ro unidirectional-utilized-bandwidth?    decimal64
         |  |  +--ro normality
         |  |     +--ro unidirectional-delay?
   performance-metric-normality
         |  |     +--ro unidirectional-min-delay?
   performance-metric-normality
         |  |     +--ro unidirectional-max-delay?
   performance-metric-normality
         |  |     +--ro unidirectional-delay-variation?
   performance-metric-normality
         |  |     +--ro unidirectional-packet-loss?
   performance-metric-normality
         |  |     +--ro unidirectional-residual-bandwidth?
   performance-metric-normality
         |  |     +--ro unidirectional-available-bandwidth?
   performance-metric-normality
         |  |     +--ro unidirectional-utilized-bandwidth?
   performance-metric-normality
         |  +--ro link-protection-type?             enumeration
         |  +--ro interface-switching-capability* [switching-
   capability]
         |  |  +--ro switching-capability              identityref
         |  |  +--ro encoding?                         identityref
         |  |  +--ro max-lsp-bandwidth* [priority]
         |  |  |  +--ro priority     uint8
         |  |  |  +--ro bandwidth?   decimal64
         |  |  +--ro packet-switch-capable
         |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
         |  |  |  +--ro interface-mtu?           uint16
         |  |  +--ro time-division-multiplex-capable
         |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
         |  |  |  +--ro indication?              enumeration
         |  |  +--ro interface-adjustment-capability* [upper-sc]
         |  |     +--ro upper-sc             identityref
```

```
         |  |     +--ro upper-encoding?      identityref
         |  |     +--ro max-lsp-bandwidth* [priority]
         |  |        +--ro priority     uint8
         |  |        +--ro bandwidth?   decimal64
         |  +--ro te-srlgs
         |     +--ro values*   te-types:srlg
        +--ro oper-status?                te-oper-status
        +--ro information-source?         enumeration
        +--ro information-source-state
        |  +--ro credibility-preference?   uint16
        |  +--ro topology
        |  |  +--ro provider-id-ref?      leafref
        |  |  +--ro client-id-ref?        leafref
        |  |  +--ro te-topology-id-ref?   leafref
        |  |  +--ro network-id-ref?       leafref
        |  +--ro routing-instance?         string
        +--ro alt-information-sources* [information-source]
        |  +--ro information-source                enumeration
        |  +--ro information-source-state
        |  |  +--ro credibility-preference?   uint16
        |  |  +--ro topology
        |  |  |  +--ro provider-id-ref?      leafref
        |  |  |  +--ro client-id-ref?        leafref
        |  |  |  +--ro te-topology-id-ref?   leafref
        |  |  |  +--ro network-id-ref?       leafref
        |  |  +--ro routing-instance?         string
        |  +--ro link-index?                       uint64
        |  +--ro administrative-group?             te-types:admin-
   groups
        |  +--ro max-link-bandwidth?               decimal64
        |  +--ro max-resv-link-bandwidth?          decimal64
        |  +--ro unreserved-bandwidth* [priority]
        |  |  +--ro priority     uint8
        |  |  +--ro bandwidth?   decimal64
        |  +--ro te-default-metric?                uint32
        |  +--ro performance-metric {te-performance-metric}?
        |  |  +--ro measurement
        |  |  |  +--ro unidirectional-delay?                 uint32
        |  |  |  +--ro unidirectional-min-delay?             uint32
        |  |  |  +--ro unidirectional-max-delay?             uint32
        |  |  |  +--ro unidirectional-delay-variation?       uint32
        |  |  |  +--ro unidirectional-packet-loss?           decimal64
        |  |  |  +--ro unidirectional-residual-bandwidth?    decimal64
        |  |  |  +--ro unidirectional-available-bandwidth?   decimal64
        |  |  |  +--ro unidirectional-utilized-bandwidth?    decimal64
        |  |  +--ro normality
```

```
      |  |     +--ro unidirectional-delay?
   performance-metric-normality
      |  |     +--ro unidirectional-min-delay?
   performance-metric-normality
      |  |     +--ro unidirectional-max-delay?
   performance-metric-normality
      |  |     +--ro unidirectional-delay-variation?
   performance-metric-normality
      |  |     +--ro unidirectional-packet-loss?
   performance-metric-normality
      |  |     +--ro unidirectional-residual-bandwidth?
   performance-metric-normality
      |  |     +--ro unidirectional-available-bandwidth?
   performance-metric-normality
      |  |     +--ro unidirectional-utilized-bandwidth?
   performance-metric-normality
      |  +--ro link-protection-type?             enumeration
      |  +--ro interface-switching-capability* [switching-
   capability]
      |  |  +--ro switching-capability              identityref
      |  |  +--ro encoding?                         identityref
      |  |  +--ro max-lsp-bandwidth* [priority]
      |  |  |  +--ro priority     uint8
      |  |  |  +--ro bandwidth?   decimal64
      |  |  +--ro packet-switch-capable
      |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
      |  |  |  +--ro interface-mtu?           uint16
      |  |  +--ro time-division-multiplex-capable
      |  |  |  +--ro minimum-lsp-bandwidth?   decimal64
      |  |  |  +--ro indication?              enumeration
      |  |  +--ro interface-adjustment-capability* [upper-sc]
      |  |     +--ro upper-sc             identityref
      |  |     +--ro upper-encoding?      identityref
      |  |     +--ro max-lsp-bandwidth* [priority]
      |  |        +--ro priority     uint8
      |  |        +--ro bandwidth?   decimal64
      |  +--ro te-srlgs
      |     +--ro values*   te-types:srlg
      +--ro recovery
      |  +--ro restoration-status?   te-recovery-status
      |  +--ro protection-status?    te-recovery-status
      +--ro underlay {te-topology-hierarchy}?
         +--ro dynamic?     boolean
         +--ro committed?   boolean
```

6. TE Topology Yang Module

```
<CODE BEGINS>

module ietf-te-topology {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-te-topology";
  // replace with IANA namespace when assigned

  prefix "tet";

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-schedule {
    prefix "sch";
  }

  import ietf-te-types {
    prefix "te-types";
  }

  import ietf-network {
    prefix "nw";
  }

  import ietf-network-topology {
    prefix "nt";
  }

  organization "TBD";
  contact "TBD";
  description "TE topology model";

  revision "2015-10-16" {
    description "Initial revision";
    reference "TBD";
  }

  /*
   * Features
   */

  feature configuration-schedule {
    description
```

```
        "This feature indicates that the system supports
         configuration scheduling.";
    }

    feature te-topology-hierarchy {
      description
        "This feature indicates that the system allows underlay
         and/or overlay TE topology hierarchy.";
    }

    feature te-performance-metric {
      description
        "This feature indicates that the system supports
         TE performance metric defined in
         RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
    }

    feature template {
      description
        "This feature indicates that the system supports
         template configuration.";
    }

    /*
     * Typedefs
     */
    typedef performance-metric-normality {
      type enumeration {
        enum "unknown" {
          value 0;
          description
            "Unknown.";
        }
        enum "normal" {
          value 1;
          description
            "Normal.";
        }
        enum "abnormal" {
          value 2;
          description
            "Abnormal. The anomalous bit is set.";
        }
      }
      description
        "Indicates whether a performance metric is normal, abnormal,
```

```
          or unknown.";
      reference
        "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
    }

    typedef te-admin-status {
      type enumeration {
        enum up {
          description
            "Enabled.";
        }
        enum down {
          description
            "Disabled.";
        }
        enum testing {
          description
            "In some test mode.";
        }
        enum preparing-maintenance {
          description
            "Resource is disabled in the control plane to prepare for
             graceful shutdown for maintenance purposes.";
          reference
            "RFC5817: Graceful Shutdown in MPLS and Generalized MPLS
             Traffic Engineering Networks";
        }
        enum maintenance {
          description
            "Resource is disabled in the data plane for maintenance
             purposes.";
        }
      }
      description
        "Defines a type representing the administrative status of
         a TE resource.";
    }

    typedef te-global-id {
      type uint32;
      description
        "An identifier to uniquely identify an operator, which can be
         either a provider or a client.
         The definition of this type is taken from RFC6370 and
         RFC5003. This attribute type is used solely to provide a
         globally unique context for TE topologies.";
```

```
    }

    typedef te-link-access-type {
      type enumeration {
        enum point-to-point {
          description
            "The link is point-to-point.";
        }
        enum multi-access {
          description
            "The link is multi-access, including broacast and NBMA.";
        }
      }
      description
        "Defines a type representing the access type of a TE link.";
      reference
        "RFC3630: Traffic Engineering (TE) Extensions to OSPF
         Version 2.";
    }

    typedef te-node-id {
      type inet:ip-address;
      description
        "An identifier for a node in a topology.
         The identifier is represented as an IPv4 or IPv6 address.
         This attribute is mapped to Router ID in
         RFC3630, RFC5329, RFC5305, and RFC 6119.";
    }

    typedef te-oper-status {
      type enumeration {
        enum up {
          description
          "Operational up.";
        }
        enum down {
          description
          "Operational down.";
        }
        enum testing {
          description
          "In some test mode.";
        }
        enum unknown {
          description
          "Status cannot be determined for some reason.";
```

```
        }
        enum preparing-maintenance {
          description
            "Resource is disabled in the control plane to prepare for
             graceful shutdown for maintenance purposes.";
          reference
            "RFC5817: Graceful Shutdown in MPLS and Generalized MPLS
             Traffic Engineering Networks";
        }
        enum maintenance {
          description
            "Resource is disabled in the data plane for maintenance
             purposes.";
        }
      }
      description
        "Defines a type representing the operational status of
         a TE resource.";
    }

    typedef te-recovery-status {
      type enumeration {
        enum normal {
          description
            "Both the recovery and working spans are fully
             allocated and active, data traffic is being
             transported over (or selected from) the working
             span, and no trigger events are reported.";
        }
        enum recovery-started {
          description
            "The recovery action has been started, but not
             completed.";
        }
        enum recovery-succeeded {
          description
            "The recovery action has succeeded. The working span has
             reported a failure/degrade condition and the user traffic
             is being transported (or selected) on the recovery
             span.";
        }
        enum recovery-failed {
          description
            "The recovery action has failed.";
        }
        enum reversion-started {
```

```
          description
            "The reversion has started.";
        }
        enum reversion-failed {
          description
            "The reversion has failed.";
        }
        enum recovery-unavailable {
          description
            "The recovery is unavailable -- either as a result of an
             operator Lockout command or a failure condition detected
             on the recovery span.";
        }
        enum recovery-admin {
          description
            "The operator has issued a command switching the user
             traffic to the recovery span.";
        }
        enum wait-to-restore {
          description
            "The recovery domain is recovering from a failuer/degrade
             condition on the working span that is being controlled by
             the Wait-to-Restore (WTR) timer.";
        }
      }
      description
        "Defines the status of a recovery action.";
      reference
        "RFC4427: Recovery (Protection and Restoration) Terminology
         for Generalized Multi-Protocol Label Switching (GMPLS).
         RFC6378: MPLS Transport Profile (MPLS-TP) Linear Protection";
    }

    typedef te-template-name {
      type string {
        pattern '/?([a-zA-Z0-9\-_.]+)(/[a-zA-Z0-9\-_.]+)*';
      }
      description
        "A type for the name of a TE node template or TE link
         template.";
    }

    typedef te-topology-event-type {
      type enumeration {
        enum "add" {
          value 0;
```

```
          description
            "A TE node or te-link has been added.";
        }
        enum "remove" {
          value 1;
          description
            "A TE node or te-link has been removed.";
        }
        enum "update" {
          value 2;
          description
            "A TE node or te-link has been updated.";
        }
      }
      description "TE  Event type for notifications";
    } // te-topology-event-type

    typedef te-topology-id {
      type string {
        pattern '/?([a-zA-Z0-9\-_.]+)(/[a-zA-Z0-9\-_.]+)*';
      }
      description
        "An identifier for a topology.";
    }

    typedef te-tp-id {
      type union {
        type uint32;          // Unnumbered
        type inet:ip-address; // IPv4 or IPv6 address
      }
      description
        "An identifier for a TE link endpoint on a node.
         This attribute is mapped to local or remote link identifier
         in RFC3630 and RFC5305.";
    }

    /*
     * Identities
     */

    identity te-flag-base {
      description "Base type for flags.";
    }

    identity te-link-flag {
      base "te-flag-base";
```

```
      description "TE link flag.";
    }

    identity te-node-flag {
      base "te-flag-base";
      description "TE node flag.";
    }

    identity te-undefined-flag {
      base "te-flag-base";
      description "Undefined flag.";
    }

    typedef te-node-flag {
      type identityref {
        base "te-node-flag";
      }
      description "Type for TE node flags.";
    }

    typedef te-link-flag {
      type identityref {
        base "te-link-flag";
      }
      description "Type for TE link flags.";
    }

    /*
     * Groupings
     */
    grouping network-ref {
      description
        "Contains the information necessary to reference a network,
         for example an underlay network.";
      leaf network-ref {
        type leafref {
          path "/nw:network/nw:network-id";
        }
        description
          "Used to reference a network, for example an underlay
           network.";
      }
    }

    grouping node-ref {
      description
```

```
        "Contains the information necessary to reference a node.";
      leaf node-ref {
        type leafref {
          path "/nw:network[nw:network-id=current()/../network-ref]"+
            "/nw:node/nw:node-id";
        }
        description
          "Used to reference a node.
           Nodes are identified relative to the network they are
           contained in.";
      }
      uses network-ref;
    }

    grouping link-ref {
      description
        "References a link in a specific network.";
      leaf link-ref {
        type leafref {
          path "/nw:network[nw:network-id=current()/../"+
            "network-ref]/nt:link/nt:link-id";
        }
        description
          "A type for an absolute reference a link instance.
           (This type should not be used for relative references.
           In such a case, a relative path should be used instead.)";
      }
      uses network-ref;
    }

    grouping tp-ref {
      description
        "References a termination point in a specific node.";
      leaf tp-ref {
        type leafref {
          path "/nw:network[nw:network-id=current()/../"+
            "network-ref]/nw:node[nw:node-id=current()/../"+
            "node-ref]/nt:termination-point/nt:tp-id";
        }
        description
          "A type for an absolute reference to a termination point.
           (This type should not be used for relative references.
           In such a case, a relative path should be used instead.)";
      }
      uses node-ref;
    }
```

```
     grouping information-source-attributes {
       description
         "The attributes identifying source that has provided the
          related information, and the source credibility.";
       leaf information-source {
         type enumeration {
           enum "unknown" {
             description "The source is unknown.";
           }
           enum "locally-configured" {
             description "Configured entity.";
           }
           enum "ospfv2" {
             description "OSPFv2.";
           }
           enum "ospfv3" {
             description "OSPFv3.";
           }
           enum "isis" {
             description "ISIS.";
           }
           enum "system-processed" {
             description "System processed entity.";
           }
           enum "other" {
             description "Other source.";
           }
         }
         description
           "Indicates the source of the information.";
       }
       container information-source-state {
         description
           "The container contains state attributes related to
            the information source.";
         leaf credibility-preference {
           type uint16;
           description
             "The preference value to calculate the traffic
              engineering database credibility value used for
              tie-break selection between different
              information-source values.
              Higher value is more preferable.";
         }
         container topology {
```

```
          description
            "When the information is processed by the system,
             the attributes in this container indicate which topology
             is used to process to generate the result information.";
          uses te-topology-ref;
        } // topology
        leaf routing-instance {
          type string;
          description
            "When applicable, this is the name of a routing instance
             from which the information is learned.";
        } // routing-information
      }
    } // information-source-attributes

    grouping performance-metric-attributes {
      description
        "Link performance information in real time.";
      reference
        "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
      leaf unidirectional-delay {
        type uint32 {
          range 0..16777215;
        }
        description "Delay or latency in micro seconds.";
      }
      leaf unidirectional-min-delay {
        type uint32 {
          range 0..16777215;
        }
        description "Minimum delay or latency in micro seconds.";
      }
      leaf unidirectional-max-delay {
        type uint32 {
          range 0..16777215;
        }
        description "Maximum delay or latency in micro seconds.";
      }
      leaf unidirectional-delay-variation {
        type uint32 {
          range 0..16777215;
        }
        description "Delay variation in micro seconds.";
      }
      leaf unidirectional-packet-loss {
        type decimal64 {
```

```
          fraction-digits 6;
          range "0 .. 50.331642";
        }
        description
          "Packet loss as a percentage of the total traffic sent
           over a configurable interval. The finest precision is
           0.000003%.";
      }
      leaf unidirectional-residual-bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Residual bandwidth that subtracts tunnel
           reservations from Maximum Bandwidth (or link capacity)
           [RFC3630] and provides an aggregated remainder across QoS
           classes.";
      }
      leaf unidirectional-available-bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Available bandwidth that is defined to be residual
           bandwidth minus the measured bandwidth used for the
           actual forwarding of non-RSVP-TE LSP packets.  For a
           bundled link, available bandwidth is defined to be the
           sum of the component link available bandwidths.";
      }
      leaf unidirectional-utilized-bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Bandwidth utilization that represents the actual
           utilization of the link (i.e. as measured in the router).
           For a bundled link, bandwidth utilization is defined to
           be the sum of the component link bandwidth
           utilizations.";
      }
    } // performance-metric-attributes

    grouping performance-metric-normality-attributes {
      description
        "Link performance metric normality attributes.";
      reference
```

```
        "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
      leaf unidirectional-delay {
        type performance-metric-normality;
        description "Delay normality.";
      }
      leaf unidirectional-min-delay {
        type performance-metric-normality;
        description "Minimum delay or latency normality.";
      }
      leaf unidirectional-max-delay {
        type performance-metric-normality;
        description "Maximum delay or latency normality.";
      }
      leaf unidirectional-delay-variation {
        type performance-metric-normality;
        description "Delay variation normality.";
      }
      leaf unidirectional-packet-loss {
        type performance-metric-normality;
        description "Packet loss normality.";
      }
      leaf unidirectional-residual-bandwidth {
        type performance-metric-normality;
        description "Residual bandwidth normality.";
      }
      leaf unidirectional-available-bandwidth {
        type performance-metric-normality;
        description "Available bandwidth normality.";
      }
      leaf unidirectional-utilized-bandwidth {
        type performance-metric-normality;
        description "Bandwidth utilization normality.";
      }
    } // performance-metric-normality-attributes

    grouping performance-metric-throttle-container {
      description
        "A container controlling performance metric throttle.";
      container performance-metric-throttle {
        if-feature te-performance-metric;
        must "suppression-interval >= measure-interval" {
          error-message
            "suppression-interval cannot be less then
             measure-interval.";
          description
            "Constraint on suppression-interval and
```

```
            measure-interval.";
          }
          description
            "Link performance information in real time.";
          reference
            "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
          leaf unidirectional-delay-offset {
            type uint32 {
              range 0..16777215;
            }
            description
              "Offset value to be added to the measured delay value.";
          }
          leaf measure-interval {
            type uint32;
            default 30;
            description
              "Interval in seconds to measure the extended metric
               values.";
          }
          leaf advertisement-interval {
            type uint32;
            description
              "Interval in seconds to advertise the extended metric
               values.";
          }
          leaf suppression-interval {
            type uint32 {
              range "1 .. max";
            }
            default 120;
            description
              "Interval in seconds to suppress advertising the extended
               metric values.";
          }
          container threshold-out {
            uses performance-metric-attributes;
            description
              "If the measured parameter falls outside an upper bound
               for all but the min delay metric (or lower bound for
               min-delay metric only) and the advertised value is not
               already outside that bound, anomalous announcement will
               be triggered.";
          }
          container threshold-in {
            uses performance-metric-attributes;
```

```
          description
            "If the measured parameter falls inside an upper bound
             for all but the min delay metric (or lower bound for
             min-delay metric only) and the advertised value is not
             already inside that bound, normal (anomalous-flag
             cleared) announcement will be triggered.";
        }
        container threshold-accelerated-advertisement {
          description
            "When the difference between the last advertised value and
             current measured value exceed this threshold, anomalous
             announcement will be triggered.";
          uses performance-metric-attributes;
        }
      }
    } // performance-metric-throttle-container

    grouping te-link-augment {
      description
        "Augmentatin for TE link.";

      container te {
        presence "TE support.";
        description
          "Indicates TE support.";

        container config {
          description
            "Configuration data.";
          uses te-link-config;
        } // config
        container state {
          config false;
          description
            "Operational state data.";
          uses te-link-config;
          uses te-link-state-derived;
        } // state
      } // te
    } // te-link-augment

    grouping te-link-config {
      description
        "TE link configuration grouping.";
      choice bundle-stack-level {
        description
```

```
          "The TE link can be partitioned into bundled
           links, or component links.";
        case bundle {
          container bundled-links {
            description
              "A set of bundled links.";
            reference
              "RFC4201: Link Bundling in MPLS Traffic Engineering
              (TE).";
            list bundled-link {
              key "sequence";
              description
                "Specify a bundled interface that is
                 further partitioned.";
              leaf sequence {
                type uint32;
                description
                  "Identify the sequence in the bundle.";
              }
              leaf src-tp-ref {
                type leafref {
                  path "../../../../../../nw:node[nw:node-id = "
                    + "current()/../../../../../nt:source/"
                    + "nt:source-node]/"
                    + "nt:termination-point/nt:tp-id";
                  require-instance true;
                }
                description
                  "Reference to another TE termination point on the
                   same souruce node.";
              }
              leaf des-tp-ref {
                type leafref {
                  path "../../../../../../nw:node[nw:node-id = "
                    + "current()/../../../../../nt:destination/"
                    + "nt:dest-node]/"
                    + "nt:termination-point/nt:tp-id";
                  require-instance true;
                }
                description
                  "Reference to another TE termination point on the
                   same destination node.";
              }
            } // list bundled-link
          }
        }
```

```
        case component {
          container component-links {
            description
              "A set of component links";
            list component-link {
              key "sequence";
              description
                "Specify a component interface that is
                 sufficient to unambiguously identify the
                 appropriate resources";

              leaf sequence {
                type uint32;
                description
                  "Identify the sequence in the bundle.";
              }
              leaf src-interface-ref {
                type string;
                description
                  "Reference to component link interface on the
                   source node.";
              }
              leaf des-interface-ref {
                type string;
                description
                  "Reference to component link interface on the
                   destinatioin node.";
              }
            }
          }
        }
      } // bundle-stack-level

      leaf-list te-link-template {
        if-feature template;
        type leafref {
          path "../../../../te/templates/link-template/name";
        }
        description
          "The reference to a TE link template.";
      }
      uses te-link-config-attributes;
    } // te-link-config

    grouping te-link-config-attributes {
      description
```

```
        "Link configuration attributes in a TE topology.";
      container te-link-attributes {
        description "Link attributes in a TE topology.";
        uses sch:schedules;
        leaf access-type {
          type te-link-access-type;
          description
            "Link access type, which can be point-to-point or
             multi-access.";
        }
        leaf-list flag {
          type te-link-flag;
          description "Link flags.";
        }
        leaf is-abstract {
          type empty;
          description "Present if the link is abstract.";
        }
        leaf name {
          type string;
          description "Link Name.";
        }
        container underlay {
          if-feature te-topology-hierarchy;
          presence
            "Indicates the underlay exists for this link.";
          description "Attributes of the te-link underlay.";
          reference
            "RFC4206: Label Switched Paths (LSP) Hierarchy with
             Generalized Multi-Protocol Label Switching (GMPLS)
             Traffic Engineering (TE)";

          uses te-link-underlay-attributes;
        } // underlay
        leaf admin-status {
          type te-admin-status;
          description
            "The administrative state of the link.";
        }

        uses performance-metric-throttle-container;
        uses te-link-info-attributes;
      }
    } // te-link-config-attributes

    grouping te-link-info-attributes {
```

```
      description
        "Advertised TE information attributes.";
      leaf link-index {
        type uint64;
        description
          "The link identifier.  If OSPF is used, this represents an
           ospfLsdbID.  If IS-IS is used, this represents an
           isisLSPID. If a locally configured link is used, this
           object represents a unique value, which is locally defined
           in a router.";
      }
      leaf administrative-group {
        type te-types:admin-groups;
        description
          "Administrative group or color of the link.
           This attribute covers both administrative group (defined in
           RFC3630, RFC5329, and RFC5305), and extended administrative
           group (defined in RFC7308).";
      }
      leaf max-link-bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Maximum bandwidth that can be seen on this link in this
           direction. Units in bytes per second.";
        reference
          "RFC3630: Traffic Engineering (TE) Extensions to OSPF
           Version 2.
           RFC5305: IS-IS Extensions for Traffic Engineering.";
      }
      leaf max-resv-link-bandwidth {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Maximum amount of bandwidth that can be reserved in this
           direction in this link. Units in bytes per second.";
        reference
          "RFC3630: Traffic Engineering (TE) Extensions to OSPF
           Version 2.
           RFC5305: IS-IS Extensions for Traffic Engineering.";
      }
      list unreserved-bandwidth {
        key "priority";
        max-elements "8";
```

```
          description
            "Unreserved bandwidth for 0-7 priority levels. Units in
             bytes per second.";
          reference
            "RFC3630: Traffic Engineering (TE) Extensions to OSPF
             Version 2.
             RFC5305: IS-IS Extensions for Traffic Engineering.";
          leaf priority {
            type uint8 {
              range "0..7";
            }
            description "Priority.";
          }
          leaf bandwidth {
            type decimal64 {
              fraction-digits 2;
            }
            description
              "Unreserved bandwidth for this level.";
          }
        }
        leaf te-default-metric {
          type uint32;
          description
            "Traffic Engineering Metric.";
        }
        container performance-metric {
          if-feature te-performance-metric;
          description
            "Link performance information in real time.";
          reference
            "RFC7471: OSPF Traffic Engineering (TE) Metric Extensions.";
          container measurement {
            description
              "Measured performance metric values. Static configuration
               and manual overrides of these measurements are also
               allowed.";
            uses performance-metric-attributes;
          }
          container normality
          {
            description
              "Performance metric normality values.";
            uses performance-metric-normality-attributes;
          }
        }
```

```
        leaf link-protection-type {
          type enumeration {
            enum "unprotected" {
              description "Unprotected.";
            }
            enum "extra-traffic" {
              description "Extra traffic.";
            }
            enum "shared" {
              description "Shared.";
            }
            enum "1-for-1" {
              description "One for one protection.";
            }
            enum "1-plus-1" {
              description "One plus one protection.";
            }
            enum "enhanced" {
              description "Enhanced protection.";
            }
          }
          description
            "Link Protection Type desired for this link.";
          reference
            "RFC4202: Routing Extensions in Support of
             Generalized Multi-Protocol Label Switching (GMPLS).";
        }
        list interface-switching-capability {
          key "switching-capability";
          description
            "List of Interface Switching Capabilities Descriptors (ISCD)
             for this link.";
          reference
            "RFC3471: Generalized Multi-Protocol Label Switching (GMPLS)
             Signaling Functional Description.
             RFC4203: OSPF Extensions in Support of Generalized
             Multi-Protocol Label Switching (GMPLS).";
          leaf switching-capability {
            type identityref {
              base te-types:switching-capabilities;
            }
            description
              "Switching Capability for this interface.";
          }
          leaf encoding {
            type identityref {
```

```
            base te-types:lsp-encoding-types;
          }
          description
            "Encoding supported by this interface.";
        }
        list max-lsp-bandwidth {
          key "priority";
          max-elements "8";
          description
            "Maximum LSP Bandwidth at priorities 0-7.";
          leaf priority {
            type uint8 {
              range "0..7";
            }
            description "Priority.";
          }
          leaf bandwidth {
            type decimal64 {
              fraction-digits 2;
            }
            description
              "Max LSP Bandwidth for this level";
          }
        }
        container packet-switch-capable {
          when "../switching-capability = 'PSC-1'  or "
            +"../switching-capability = 'PSC-2' or "
            +"../switching-capability = 'PSC-3' or "
            +"../switching-capability = 'PSC-4'" {
            description "Valid only for PSC";
          }
          description
            "Interface has packet-switching capabilities.";
          leaf minimum-lsp-bandwidth {
            type decimal64 {
              fraction-digits 2;
            }
            description
              "Minimum LSP Bandwidth. Units in bytes per second";
          }
          leaf interface-mtu {
            type uint16;
            description
              "Interface MTU.";
          }
        }
```

```
          container time-division-multiplex-capable {
            when "../switching-capability = 'TDM'" {
              description "Valid only for TDM";
            }
            description
              "Interface has time-division multiplex capabilities.";

            leaf minimum-lsp-bandwidth {
              type decimal64 {
                fraction-digits 2;
              }
              description
                "Minimum LSP Bandwidth. Units in bytes per second.";
            }
            leaf indication {
              type enumeration {
                enum "standard" {
                  description
                    "Indicates support of standard SONET/SDH.";
                }
                enum "arbitrary" {
                  description
                    "Indicates support of arbitrary SONET/SDH.";
                }
              }
              description
                "Indication whether the interface supports Standard or
                 Arbitrary SONET/SDH";
            }
          }
          list interface-adjustment-capability {
            key "upper-sc";
            description
              "List of Interface Adjustment Capability Descriptors
               (IACD)for this link.";
            reference
              "RFC6001: Generalized MPLS (GMPLS) Protocol Extensions
               for Multi-Layer and Multi-Region Networks (MLN/MRN).";
            leaf upper-sc {
              type identityref {
                base te-types:switching-capabilities;
              }
              description
                "Switching Capability for this interface.";
            }
            leaf upper-encoding {
```

```
            type identityref {
              base te-types:lsp-encoding-types;
            }
            description
              "Encoding supported by this interface.";
          }
          list max-lsp-bandwidth {
            key "priority";
            max-elements "8";
            description
              "Maximum LSP Bandwidth at priorities 0-7.";
            leaf priority {
              type uint8 {
                range "0..7";
              }
              description "Priority.";
            }
            leaf bandwidth {
              type decimal64 {
                fraction-digits 2;
              }
              description
                "Max LSP Bandwidth for this level.";
            }
          }
        } // interface-adjustment-capability
      } // interface-switching-capability
      container te-srlgs {
        description
          "A list of SLRGs.";
        leaf-list values {
          type te-types:srlg;
          description "SRLG value.";
          reference
            "RFC4202: Routing Extensions in Support of
             Generalized Multi-Protocol Label Switching (GMPLS).";
        }
      }
    } // te-link-info-attributes

    grouping te-link-state-derived {
      description
        "Link state attributes in a TE topology.";
      leaf oper-status {
        type te-oper-status;
        description
```

```
          "The current operational state of the link.";
      }
      uses information-source-attributes;
      list alt-information-sources {
        key "information-source";
        description
          "A list of information sources learned but not used.";
        uses information-source-attributes;
        uses te-link-info-attributes;
      }
      container recovery {
        description
          "Status of the recovery process.";
        leaf restoration-status {
          type te-recovery-status;
          description
            "Restoration status.";
        }
        leaf protection-status {
          type te-recovery-status;
          description
            "Protection status.";
        }
      }
      container underlay {
        if-feature te-topology-hierarchy;
        description "State attributes for te-link underlay.";
        uses te-link-state-underlay-attributes;
      }
    } // te-link-state-derived

    grouping te-link-state-underlay-attributes {
      description "State attributes for te-link underlay.";
      leaf dynamic {
        type boolean;
        description
          "true if the underlay is dynamically created.";
      }
      leaf committed {
        type boolean;
        description
          "true if the underlay is committed.";
      }
    } // te-link-state-underlay-attributes

    grouping te-link-underlay-attributes {
```

```
        description "Attributes for  te-link underlay.";
        reference
          "RFC4206: Label Switched Paths (LSP) Hierarchy with
           Generalized Multi-Protocol Label Switching (GMPLS)
           Traffic Engineering (TE)";
        container underlay-primary-path {
          description
            "The service path on the underlay topology that
             supports this link.";
          uses te-topology-ref;
          list path-element {
            key "path-element-id";
            description
              "A list of path elements describing the service path.";
            leaf path-element-id {
              type uint32;
              description "To identify the element in a path.";
            }
            uses te-path-element;
          }
        } // underlay-primary-path
        list underlay-backup-path {
          key "index";
          description
            "A list of backup service paths on the underlay topology
             that protect the underlay primary path. If the primary path
             is not protected, the list contains zero elements. If the
             primary path is protected, the list contains one or more
             elements.";
          leaf index {
            type uint32;
            description
              "A sequence number to identify a backup path.";
          }
          uses te-topology-ref;
          list path-element {
            key "path-element-id";
            description
              "A list of path elements describing the backup service
               path";
            leaf path-element-id {
              type uint32;
              description "To identify the element in a path.";
            }
            uses te-path-element;
          }
```

```
      } // underlay-backup-path
      leaf underlay-protection-type {
        type uint16;
        description
          "Underlay protection type desired for this link";
      }
      container underlay-trail-src {
        uses tet:tp-ref;
        description
          "Source TE link of the underlay trail.";
      }
      container underlay-trail-des {
        uses tet:tp-ref;
        description
          "Destination TE link of the underlay trail.";
      }
    } // te-link-underlay-attributes

    grouping te-node-augment {
      description
        "Augmentatin for TE node.";

      container te {
        presence "TE support.";
        description
          "Indicates TE support.";

        leaf te-node-id {
          type te-node-id;
          mandatory true;
          description
            "The identifier of a node in the TE topology.
             A node is specific to a topology to which it belongs.";
        }

        container config {
          description
            "Configuration data.";
          uses te-node-config;
        } // config
        container state {
          config false;
          description
            "Operational state data.";

          uses te-node-config;
```

```
          uses te-node-state-derived;
        } // state
      } // te
    } // te-node-augment

    grouping te-node-config {
      description "TE node configuration grouping.";

      leaf-list te-node-template {
        if-feature template;
        type leafref {
          path "../../../../te/templates/node-template/name";
        }
        description
          "The reference to a TE node template.";
      }
      uses te-node-config-attributes;
    } // te-node-config

    grouping te-node-config-attributes {
      description "Configuration node attributes in a TE topology.";
      container te-node-attributes {
        description "Containing node attributes in a TE topology.";
        uses sch:schedules;
        leaf admin-status {
          type te-admin-status;
          description
            "The administrative state of the link.";
        }
        list connectivity-matrix {
          key "id";
          description
            "Represents node's switching limitations, i.e. limitations
             in interconnecting network TE links across the node.";
          leaf id {
            type uint32;
            description "Identifies the connectivity-matrix entry.";
          }
          container from {
            uses tet:tp-ref;
            description
              "Reference to source NTP.";
          }
          container to {
            uses tet:tp-ref;
            description
```

```
            "Reference to destination NTP.";
          }
          leaf is-allowed {
            type boolean;
            description
              "true  - switching is allowed,
               false - switching is disallowed.";
          }
        }
        leaf domain-id {
          type uint32;
          description
            "Identifies the domain that this node belongs.
             This attribute is used to support inter-domain links.";
          reference
            "RFC5152: A Per-Domain Path Computation Method for
             Establishing Inter-Domain Traffic Engineering (TE)
             Label Switched Paths (LSPs).
             RFC5392: OSPF Extensions in Support of Inter-Autonomous
             System (AS) MPLS and GMPLS Traffic Engineering.
             RFC5316: ISIS Extensions in Support of Inter-Autonomous
             System (AS) MPLS and GMPLS Traffic Engineering.";
        }
        leaf-list flag {
          type te-node-flag;
          description "Node operational flags.";
        }
        leaf is-abstract {
          type empty;
          description
            "Present if the node is abstract, not present if the node
             is actual.";
        }
        leaf name {
          type inet:domain-name;
          description "Node name.";
        }
        leaf-list signaling-address {
          type inet:ip-address;
          description "Node signaling address.";
        }
        container underlay-topology {
          if-feature te-topology-hierarchy;
          description
            "When an abstract node encapsulates a topology,
```

```
            the attributes in this container point to said
   topology.";
          uses te-topology-ref;
        }
      }
    } // te-node-config-attributes

    grouping te-node-state-derived {
      description "Node state attributes in a TE topology.";
      leaf oper-status {
        type te-oper-status;
        description
          "The current operational state of the node.";
      }
      leaf is-multi-access-dr {
        type empty;
        description
          "The presence of this attribute indicates that this TE node
           is a pseudonode elected as a designated router.";
        reference
          "RFC3630: Traffic Engineering (TE) Extensions to OSPF
           Version 2.
           RFC1195: Use of OSI IS-IS for Routing in TCP/IP and Dual
           Environments.";
      }
      uses information-source-attributes;
    } // te-node-state-derived

    grouping te-path-element {
      description
        "A group of attributes defining an element in a TE path
        such as TE node, TE link, TE atomic resource or label.";
      uses te-types:explicit-route-subobject;
    } // te-path-element

    grouping te-termination-point-augment {
      description
        "Augmentatin for TE termination point.";

      container te {
        presence "TE support.";
        description
          "Indicates TE support.";

        leaf te-tp-id {
          type te-tp-id;
```

```
          mandatory true;
          description
            "An identifier to uniquely identify a TE termination
             point.";
        }

        container config {
          description
            "Configuration data.";
          uses te-termination-point-config;
        } // config
        container state {
          config false;
          description
            "Operational state data.";
          uses te-termination-point-config;
        } // state
      } // te
    } // te-termination-point-augment

    grouping te-termination-point-config {
      description
        "TE termination point configuration grouping.";
      uses sch:schedules;
    } // te-termination-point-config

    grouping te-topology-augment {
      description
        "Augmentatin for TE topology.";

      container te {
        presence "TE support.";
        description
          "Indicates TE support.";

        leaf provider-id {
          type te-global-id;
          mandatory true;
          description
            "An identifier to uniquely identify a provider.";
        }
        leaf client-id {
          type te-global-id;
          mandatory true;
          description
            "An identifier to uniquely identify a client.";
```

```
        }
        leaf te-topology-id {
          type te-topology-id;
          mandatory true;
          description
            "It is presumed that a datastore will contain many
             topologies. To distinguish between topologies it is
             vital to have UNIQUE topology identifiers.";
        }

        container config {
          description
            "Configuration data.";
          uses te-topology-config;
        } // config
        container state {
          config false;
          description
            "Operational state data.";
          uses te-topology-config;
        } // state

        container templates {
          description
            "Configuration parameters for templates used for TE
             topology.";

          list node-template {
            if-feature template;
            key "name";
            leaf name {
              type te-template-name;
              description
                "The name to identify a TE node template.";
            }
            description
              "The list of TE node templates used to define sharable
               and reusable TE node attributes.";
            uses template-attributes;
            uses te-node-config-attributes;
          } // node-template

          list link-template {
            if-feature template;
            key "name";
            leaf name {
```

```
              type te-template-name;
              description
                "The name to identify a TE link template.";
            }
            description
              "The list of TE link templates used to define sharable
               and reusable TE link attributes.";
            uses template-attributes;
            uses te-link-config-attributes;
          } // link-template
        } // templates
      } // te
    } // te-topology-augment

    grouping te-topology-config {
      description
        "TE topology configuration grouping.";
      uses sch:schedules;
    } // te-topology-config

    grouping te-topology-ref {
      description
        "References a TE topology.";
      leaf provider-id-ref {
        type leafref {
          path "/nw:network/tet:te/tet:provider-id";
          require-instance false;
        }
        description
          "A reference to a provider-id.";
      }
      leaf client-id-ref {
        type leafref {
          path "/nw:network/tet:te/tet:client-id";
          require-instance false;
        }
        description
          "A reference to a client-id.";
      }
      leaf te-topology-id-ref {
        type leafref {
          path "/nw:network/tet:te/tet:te-topology-id";
          require-instance false;
        }
        description
          "A reference to a te-topology-id.";
```

```
      }
      leaf network-id-ref {
        type leafref {
          path "/nw:network/nw:network-id";
          require-instance false;
        }
        description
          "A reference to a network-id in base ietf-network module.";
      }
    } // te-topology-ref

    grouping te-topology-type {
      description
        "Identifies the TE topology type.";
      container te-topology {
        presence "Indicates TE topology.";
        description
          "Its presence identifies the TE topology type.";
      }
    } // te-topology-type

    grouping template-attributes {
      description
        "Common attributes for all templates.";

      leaf priority {
        type uint16;
        description
          "The preference value to resolve conflicts between different
           templates. When two or more templates specify values for
           one configuration attribute, the value from the template
           with the highest priority is used.";
      }
      leaf reference-change-policy {
        type enumeration {
          enum no-action {
            description
              "When an attribute changes in this template, the
               configuration node referring to this template does
               not take any action.";
          }
          enum not-allowed {
            description
              "When any configuration object has a reference to this
               template, changing this template is not allowed.";
          }
```

```
          enum cascade {
            description
              "When an attribute changes in this template, the
               configuration object referring to this template applies
               the new attribute value to the corresponding
               configuration.";
          }
        }
        description
          "This attribute specifies the action taken to a
           configuration node that has a reference to this template.";
      }
    } // template-attributes

    /*
     * Configuration data nodes
     */
    augment "/nw:network/nw:network-types" {
      description
        "Introduce new network type for TE topology.";
      uses te-topology-type;
    }

    augment "/nw:network" {
      when "nw:network-types/te-topology" {
        description
          "Augmentation parameters apply only for networks with
           TE topology type.";
      }
      description
        "Configuration parameters for TE topology.";
      uses te-topology-augment;
    }

    augment "/nw:network/nw:node" {
      when "../nw:network-types/te-topology" {
        description
          "Augmentation parameters apply only for networks with
           TE topology type.";
      }
      description
        "Configuration parameters for TE at node level.";
      uses te-node-augment;
    }

    augment "/nw:network/nt:link" {
```

```
      when "../nw:network-types/te-topology" {
        description
          "Augmentation parameters apply only for networks with
           TE topology type.";
      }
      description
        "Configuration parameters for TE at link level";
      uses te-link-augment;
    }

    augment "/nw:network/nw:node/"
          +"nt:termination-point" {
      when "../../nw:network-types/te-topology" {
        description
          "Augmentation parameters apply only for networks with
           TE topology type.";
      }
      description
        "Configuration parameters for TE at termination point level";
      uses te-termination-point-augment;
    }

    /*
     * Operational state data nodes
     */

    /*
     * Notifications
     */

    notification te-node-event {
      description "Notification event for TE node.";
      leaf event-type {
        type te-topology-event-type;
        description "Event type.";
      }
      uses tet:node-ref;
      uses te-topology-type;
      uses tet:te-node-config-attributes;
      uses tet:te-node-state-derived;
    }

    notification te-link-event {
      description "Notification event for TE link.";
      leaf event-type {
        type te-topology-event-type;
```

```
        description "Event type";
      }
      uses tet:link-ref;
      uses te-topology-type;
      uses tet:te-link-config-attributes;
      uses tet:te-link-state-derived;
    }

    augment "/te-link-event/te-link-attributes/underlay" {
      description "Add state attributes to te-link underlay.";
      uses te-link-state-underlay-attributes;
    }
  }
```

<CODE ENDS>

7. Security Considerations

The transport protocol used for retrieving/manipulating the TE topology data MUST support authentication and SHOULD support encryption. The data-model by itself does not create any security implications.

8. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-te-topology
XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-te-topology
namespace: urn:ietf:params:xml:ns:yang:ietf-te-topology
prefix: tet

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

[RFC3945] Mannie, E., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", October 2004.

[YANG-NET-TOPO] Clemm, A., "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo (Work in Progress).

[YANG-PUSH] Clemm, A., "Subscribing to YANG datastore push updates", draft-clemm-netconf-yang-push (Work in Progress).

9.2. Informative References

[RFC2702] Awduche, D., "Requirements for Traffic Engineering Over MPLS", RFC 2702, September 1999.

10. Acknowledgments

The authors would like to thank Lou Berger, Sue Hares, Mazen Khaddam, Cyril Margaria and Zafar Ali for participating in design discussions and providing valuable insights.

Appendix A - Schedule Model

A.1 Tree Structure

```
module: ietf-schedule
grouping schedules:
 +--rw schedules
    +--rw schedule* [schedule-id]
       +--rw schedule-id          uint32
       +--rw start?               yang:date-and-time
       +--rw schedule-duration?   string
       +--rw repeat-interval?     string
```

A.2 YANG Module

<CODE BEGINS>

```
module ietf-schedule {
  yang-version 1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-schedule";
  // replace with IANA namespace when assigned

  prefix "sch";

  import ietf-yang-types {
    prefix "yang";
  }

  organization "TBD";
  contact "TBD";
  description
    "The model allows time scheduling parameters to be specified.";

  revision "2015-10-09" {
    description "Initial revision";
    reference "TBD";
  }

  /*
   * Groupings
   */

  grouping schedules {
    description
      "A list of schedules defining when a particular
       configuration takes effect.";
    container schedules {
      description
        "Container of a schedule list defining when a particular
         configuration takes effect.";
      list schedule {
        key "schedule-id";
        description "A list of schedule elements.";

        leaf schedule-id {
          type uint32;
          description "Identifies the schedule element.";
```

```
          }
          leaf start {
            type yang:date-and-time;
            description "Start time.";
          }
          leaf schedule-duration {
            type string {
              pattern
                'P(\d+Y)?(\d+M)?(\d+W)?(\d+D)?T(\d+H)?(\d+M)?(\d+S)?';
            }
            description "Schedule duration in ISO 8601 format.";
          }
          leaf repeat-interval {
            type string {
              pattern
                'R\d*/P(\d+Y)?(\d+M)?(\d+W)?(\d+D)?T(\d+H)?(\d+M)?'
                + '(\d+S)?';
            }
            description "Repeat interval in ISO 8601 format.";
          }
        }
      }
    } // schedules
  }

  <CODE ENDS>
```

Contributors

Sergio Belotti
Alcatel Lucent
Email: sergio.belotti@alcatel-lucent.com

Dieter Beller
Alcatel Lucent
Email: dieter.beller@alcatel-lucent.com

Authors' Addresses

Xufeng Liu
Ericsson
Email: xufeng.liu@ericsson.com

Igor Bryskin
ADVA Optical Networking

Email: ibryskin@advaoptical.com

Vishnu Pavan Beeram
Juniper Networks
Email: vbeeram@juniper.net

Tarek Saad
Cisco Systems Inc
Email: tsaad@cisco.com

Himanshu Shah
Ciena
Email: hshah@ciena.com

Oscar Gonzalez De Dios
Telefonica
Email: oscar.gonzalezdedios@telefonica.com