TEAS Working Group
Internet Draft
Intended status: Informational

Igor Bryskin
Huawei Technologies
Vishnu Pavan Beeram
Juniper Networks
Tarek Saad
Cisco Systems Inc
Xufeng Liu
Jabil

Expires: December 27, 2017 June 27, 2017

TE Topology and Tunnel Modeling for Transport Networks
draft-bryskin-te-topo-and-tunnel-modeling-00

Status of this Memo

Copyright Notice

Abstract

This document describes how to model TE topologies and tunnels for transport networks, by using the TE topology YANG model [I-D.ietf-teas-yang-te-topo] and the TE tunnel YANG model [I-D.ietf-teas-yang-te].

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. Modeling Considerations

1.1. TE Topology Model

The TE Topology Model is written in YANG modeling language. It is defined and developed by the IETF TEAS WG and is documented as “YANG Data Model for TE Topologies” [I-D.ietf-teas-yang-te-topo]. The model describes a TE network provider’s Traffic Engineering data store as it is seen by a client. It allows for the provider to convey to each of its clients:

- information on network resources available to the client in the form of one or several native TE topologies (for example, one for each layer network supported by the provider);
- one or several abstract TE topologies, customized on per-client basis and sorted according to the provider’s preference as to how the abstract TE topologies are to be used by the client;
- updates with incremental changes happened to the previously provided abstract/native TE topology elements;

- updates on telemetry/state information the client has expressed interest in;

- overlay/underlay relationships between the TE topologies provided to the client (e.g. TE path computed in an underlay TE topology supporting a TE link in an overlay TE topology);

- client/server inter-layer adaptation relationships between the TE topologies provided to the client in the form of TE inter-layer locks or transitional links;

The TE Topology Model allows a network client to:

- (Re-)configure/negotiate abstract TE topologies provided to the client by a TE network provider, so that said abstract TE topologies optimally satisfy the client's needs, constraints and optimization criteria, based on the client's network planning, service forecasts, telemetry information extracted from the network, previous history of service provisioning and performance monitoring, etc.;

- Obtain abstract/native TE topologies from multiple providers and lock them horizontally (inter-domain) and vertically (inter-layer) into the client's own native TE topologies;

- Configure, with each provider the trigger, frequency and contents of the TE topology update notifications;

- Configure, with each provider the trigger, frequency and contents of the TE topology telemetry (e.g. statistics counters) update notifications.

1.2. TE Topology Modeling Constructs

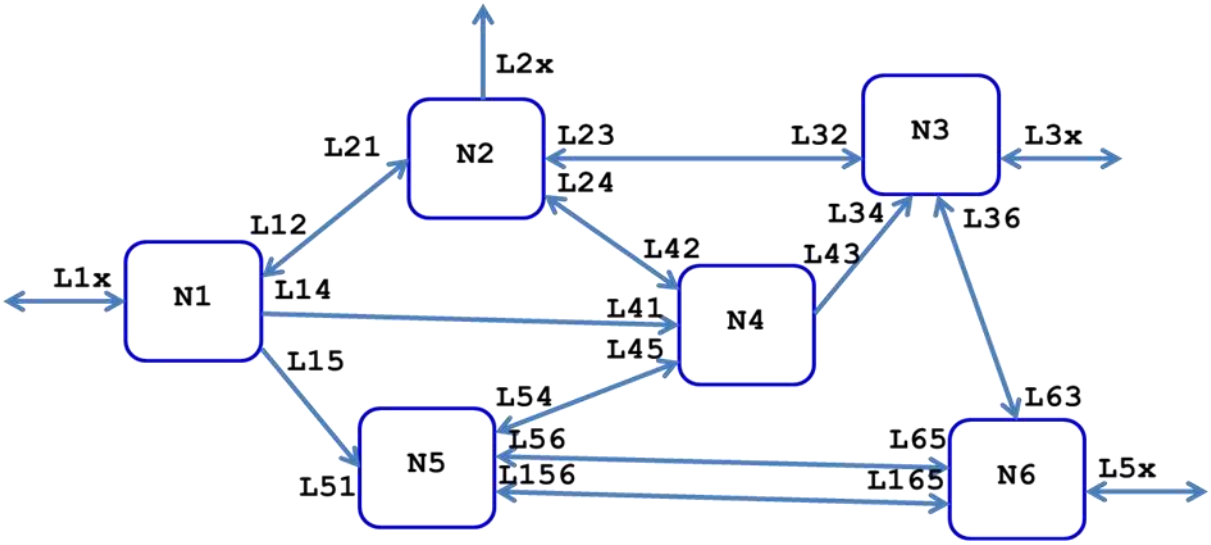

Figure 1. TE Topology

- **TE domain** - a multi-layer traffic engineered network under direct and complete control of a single authority, network provider. TE domain can be described by one or more TE topologies. For example, separate TE topologies can describe each of the domain's layer networks. TE domain can hierarchically encompass/parent other (child) TE domains, and can be encompassed by its own parent.

- **TE topology** - a graphical representation of a TE domain. TE topology is comprised of TE nodes (TE graph vertices) interconnected via TE links (TE graph edges).

```
/* TE topology */
augment /nw:networks/nw:network:
   /* TE topology global ID */
   +--rw provider-id?      te-types:te-global-id
   +--rw client-id?        te-types:te-global-id
   +--rw te-topology-id?   te-types:te-topology-id
.....................................................................
   /* TE topology general parameters */
     |  +--rw preference?               uint8
     |  +--rw optimization-criterion?   identityref
.....................................................................
```

```
          /* TE topology list of TE nodes */
   augment /nw:networks/nw:network/nw:node:
      +--rw te-node-id?   te-types:te-node-id
   .....................................................................
          /* TE topology list of TE links */
   augment /nw:networks/nw:network/nt:link:
   .....................................................................
         /* TE topology list of TE link termination points */
   augment /nw:networks/nw:network/nw:node/nt:termination-point:
      +--rw te-tp-id?   te-types:te-tp-id
   .....................................................................
```

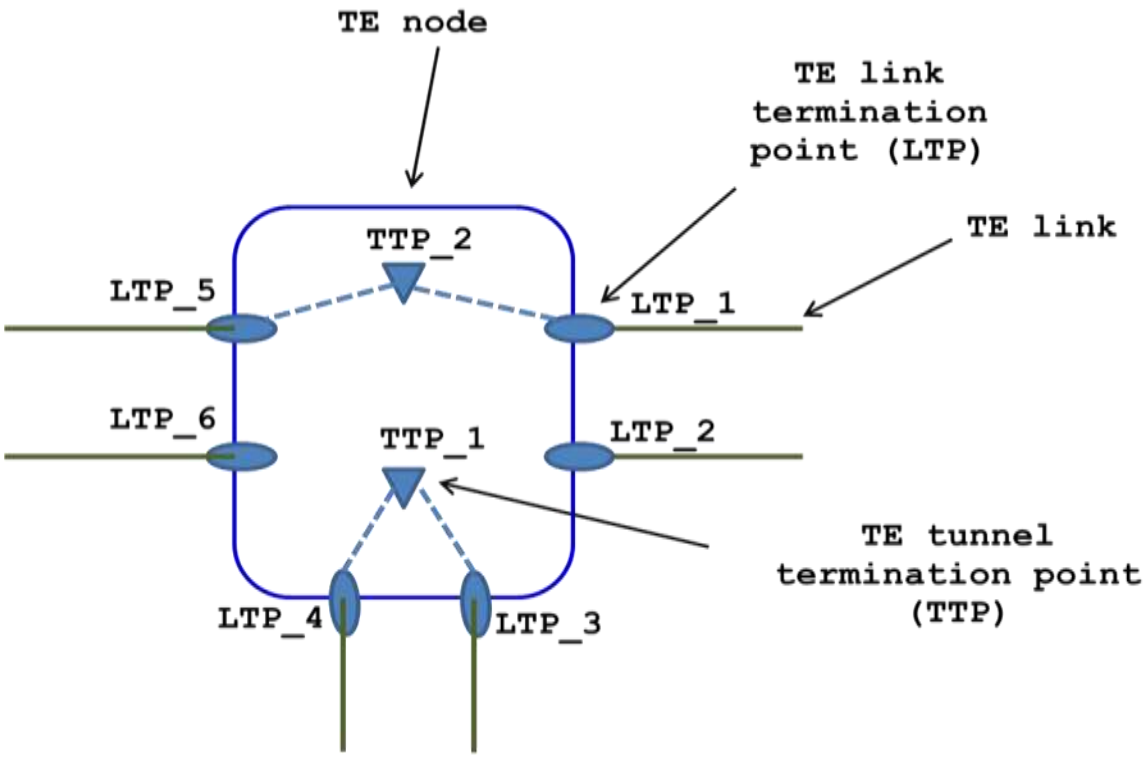

Figure 2. TE Node

- **TE node** - an element of a TE topology (appears as a vertex on TE graph). A TE node represents one or several nodes (physical switches), or a fraction of a node. A TE node belongs to and is fully defined in exactly one TE topology. A TE node is assigned a TE topology scope-unique ID. TE node attributes include information related to the data plane aspects of the associated node(s) (e.g. TE node's connectivity matrix), as well as configuration data (such as TE node name). A given TE node can be reached on the TE graph, representing the TE topology, over one of TE links terminated by the TE node.

```
/* TE node */
augment /nw:networks/nw:network/nw:node:
   /* TE node ID */
   +--rw te-node-id?   te-types:te-node-id
.....................................................................
   /* TE node general attributes */
      |  +--rw te-node-attributes */
.....................................................................
   /* TE node connectivity matrices */
      |     +--rw connectivity-matrices
.....................................................................
   /* TE node underlay TE topology */
            |     +--rw underlay-topology {te-topology-hierarchy}?
            |        +--rw network-ref?   leafref
.....................................................................
   /* TE node information sources*/
      |  +--ro information-source-entry* [information-source]
.....................................................................
  /* TE node statistics */
     +--ro statistics
.....................................................................
  /* TE node TTP list */
     +--rw tunnel-termination-point* [tunnel-tp-id]
.....................................................................
```

- **TE link** - an element of a TE topology (appears as an edge on TE graph), TE link is unidirectional and its arrow indicates the TE link's direction. Edges with two arrows on the TE topology graph (see Figure 1) represent bi-directional combinations of two parallel oppositely directed TE links. A TE link represents one or several physical links or a fraction of a physical link. A TE link belongs to and is fully defined in exactly one TE topology. A TE link is assigned a TE topology scope-unique ID. TE link attributes include parameters related to the data plane aspects of the associated link(s) (e.g. unreserved bandwidth, resource maps/pools, etc.), as well as the configuration data (such as remote node/link IDs, SRLGs, administrative colors, etc.) A TE link is connected to a TE node, terminating the TE link via exactly one TE link termination point (LTP).

```
/* TE link */
augment /nw:networks/nw:network/nt:link:
/* TE link bundle information */
      |  +--rw (bundle-stack-level)?
      |  |  |  +--rw bundled-links
      |  |     +--rw component-links
.....................................................................
/* TE link general attributes */
    |  +--rw te-link-attributes

.....................................................................
/* TE link underlay TE topology */
      |     +--rw underlay! {te-topology-hierarchy}?
      |     |  +--rw primary-path
      |     |  +--rw backup-path* [index]

.....................................................................
/* TE link layer network */
     |     +--rw interface-switching-capability* [switching-
capability encoding]

.....................................................................
/* TE link protection type */
     |     |  +--rw protection-type?   uint16

.....................................................................

/* TE link supporting TE tunnels */
    |     |  +--rw tunnels
```

```
   .....................................................................
   /* TE link transitional link flag */
        |  +--ro is-transitional?            empty


   .....................................................................
   /* TE link information sources */
         |  +--ro information-source?         te-info-source

   .....................................................................
   /* TE link statistics */
        +--ro statistics

   .....................................................................
```

- **Intra-domain TE link** - TE link connecting two TE nodes within the same TE topology representing a TE network domain (e.g. L14 in Figure 1). From the point of view of the TE topology where the intra-domain TE link is defined, the TE link is close-ended, that is, both local and remote TE nodes of the link are defined in the same TE topology.

- **Inter-domain TE link** - TE link connecting two border TE nodes that belong to separate TE topologies describing neighboring TE network domains (e.g. L3x in Figure 1). From the point of view of the TE topology where the inter-domain TE link is defined, the TE link is open-ended, that is, the remote TE node of the link is not defined in the TE topology where the local TE node and the TE link itself are defined.

  [Note: from the point of view of a TE node terminating an inter-domain TE link there is no difference between inter-domain and access TE links]

- **Access TE link** - TE link connecting a border TE node of a TE topology describing a TE network domain to a TE node of a TE topology describing a customer network site (e.g. L1x in Figure 1) From the point of view of the TE topology where the access TE link is defined, the TE link is open-ended, that is, the remote TE node of the link (t.e. TE node representing customer network element(s)) is not defined in the TE topology where the local TE node and the TE link itself are defined.

[Note: from the point of view of a TE node terminating an access TE link there is no difference between access and inter-domain TE links]

- **Dynamic TE link** - a TE link that shows up in (and disappears from) a TE topology as a result of multi-layer traffic engineering. Dynamic TE link (supported by a hierarchy TE tunnel dynamically set up in a server layer network) is automatically (i.e. without explicit configuration request) added to a client layer network TE topology to augment the topology with additional flexibility to ensure successful completion of the path computation for and provisioning of a client layer network connection/LSP. For example, an ODUk hierarchy TE tunnel can support a dynamic Ethernet layer TE link to enable provisioning of an Ethernet layer connection on a network that does not have sufficient static Ethernet layer connectivity. Likewise, dynamic TE link is automatically removed from the TE topology (and its supporting hierarchy TE tunnel released) as soon as the TE link stops carrying client layer connections/LSPs.

- **TE link termination point (LTP)** - a conceptual point of connection of a TE node to one of the TE links terminated by the TE node (see Figure 2a). Unlike TE link, LTP is bi-directional – an inbound TE link and an oppositely directed outbound TE link have to be connected to the TE node via the same LTP to constitute a bi-directional TE link combination.

Figure 2a. Bi-directional TE link combination (left), independent uni-directional TE links (right)

```
/* LTP */
augment /nw:networks/nw:network/nw:node/nt:termination-point:
/* LTP ID */
   +--rw te-tp-id?   te-types:te-tp-id
/* LTP network layer ID */
      |  +--rw interface-switching-capability* [switching-
capability encoding]
      |  |  +--rw switching-capability    identityref
```

```
      |  |  +--rw encoding                identityref
   /* LTP bandwidth information */
      |  |  +--rw max-lsp-bandwidth* [priority]
      |  |     +--rw priority     uint8
      |  |     +--rw bandwidth?   te-bandwidth
   /* LTP inter-layer locks */
      |  +--rw inter-layer-lock-id?              uint32

   ...............................................................
```

- **TE tunnel termination point (TTP)** - an element of TE topology representing one or several potential TE tunnel termination/adaptation points (e.g. OCh layer transponder). A TTP is hosted by exactly one TE node (see Figure 2). A TTP is assigned a TE node scope-unique ID. Depending on the TE node's internal constraints, a given TTP hosted by the TE node could be accessed via one, several or all TE links originated/terminated from/by the TE node. TTP's important attributes include Local Link Connectivity List, Adaptation Client Layer List, TE inter-layer locks (see below), Unreserved Adaptation Bandwidth (announcing the TTP's remaining adaptation resources sharable between all potential client LTPs), and Property Flags (indicating miscellaneous properties of the TTP, such as capability to support 1+1 protection for a TE tunnel terminated on the TTP).

```
   /* TTP */
        +--rw tunnel-termination-point* [tunnel-tp-id]
   /* TTP ID */
           +--rw tunnel-tp-id                              binary
   /* TTP layer network ID */
           |  +--rw switching-capability?        identityref
           |  +--rw encoding?                    identityref
   //* Inter-layer-locks supported by TTP */
           |  +--rw inter-layer-lock-id?         uint32
   /* TTP's protection capabilities */
           |  +--rw protection-type?             identityref
   /* TTP's list of client layer users */
           |  +--rw client-layer-adaptation

   ...............................................................
   /* TTP's Local Link Connectivity List (LLCL) */
```

```
            |  +--rw local-link-connectivities
```

.....

- **Label** – in the context of circuit switched layer networks identifies a particular resource on a TE link (e.g. Och wavelength, ODUk container)

```
+--:(label)
   +--rw value?   rt-types:generalized-label
```

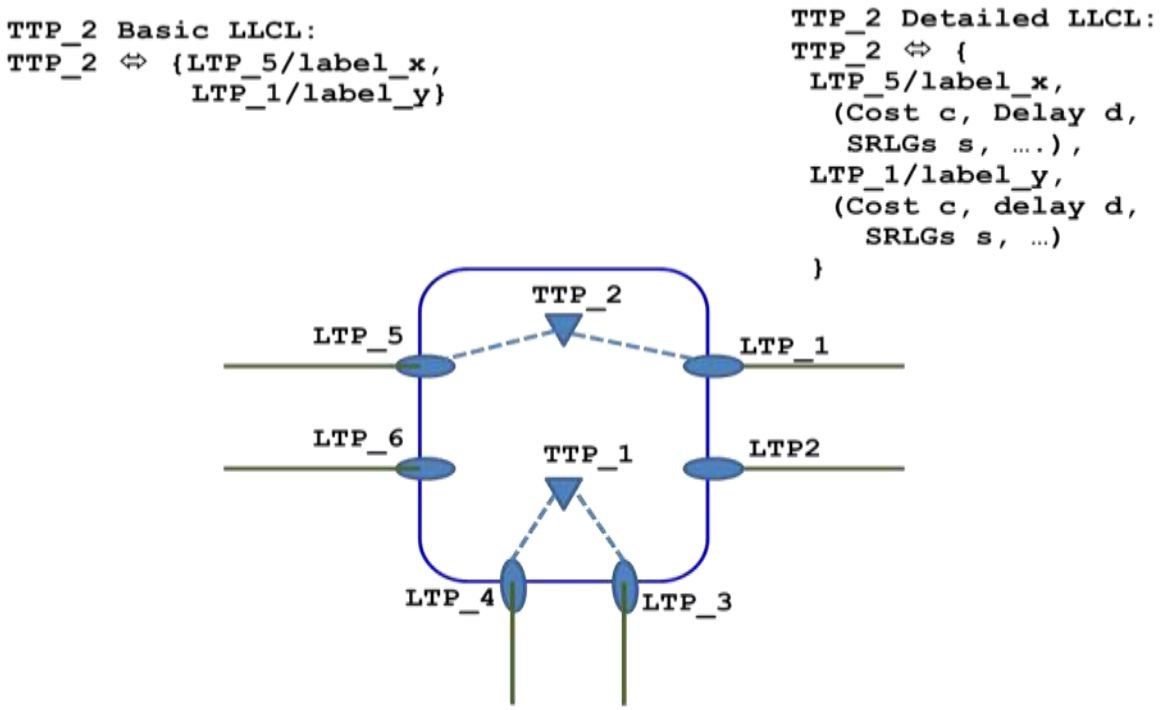

Figure 3. TTP Local Link Connectivity List

- **TTP basic local link connectivity list (basic LLCL)** – a list of TE link/label combinations terminated by the TTP-hosting TE node (effectively the same as LTP/label pairs), which the TTP could be connected to (see Figure 3, upper left). From the point of view of a potential TE path, basic LLCL provides a list of permissible LTP/label pairs the TE path needs to start/stop on for a connection, taking the TE path, to be successfully terminated on the TTP in question.

- **TTP detailed local link connectivity list (detailed LLCL)** – basic LLCL extended to provide a set of costs (such as intra-node summary TE metric, delay, SRLGs, etc.) associated with each LLCL entry (see Figure 3, upper right)

```
   /* TTP LLCL */
   |  +--rw local-link-connectivities
       |     +--rw number-of-entries?         uint16
       /* LLCL entry */

       /* LLCL entry LTP */
       |        +--rw link-tp-ref                leafref

   ....................................................................

   /* LLC entry label range */
   |     +--rw label-restriction* [inclusive-exclusive label-start]
   |     |  +--rw inclusive-exclusive    enumeration
   |     |  +--rw label-start            rt-types:generalized-label
         |     |  +--rw label-end?             rt-types:generalized-
   label
         |     |  +--rw range-bitmap?          binary

   /* LLCL entry underlay TE path(s) */
   |     +--rw underlay! {te-topology-hierarchy}?
   |     |  +--rw primary-path
   |     |  +--rw backup-path* [index]
   /* LLCL entry protection type */
   |     |  +--rw protection-type?   uint16
   /* LLCL entry supporting TE tunnels */
   |     |  +--rw tunnels
   /* LLCL entry bandwidth parameters */
   |     +--rw max-lsp-bandwidth* [priority]

   ....................................................................
```

```
   /* LLCL entry metrics  (vector of costs) */
   |     +--rw te-default-metric?         uint32
   |     +--rw te-delay-metric?           uint32
   |     +--rw te-srlgs
   |     |  +--rw value*   te-types:srlg
   |     +--rw te-nsrlgs {nsrlg}?

   ..............................................................
   /* LLCL entry ID */
   |     |  +--rw id*   uint32
```

- **TTP adaptation client layer list** - a list of client layers that could be directly adopted by the TTP. This list is necessary to describe complex multi-layer (more than two layer) client-server layer hierarchies and, in particular, to identify the position of the TTP in said hierarchies.

```
   /* TTP adaptation client layer list */
            |  +--rw client-layer-adaptation
            |  |  +--rw switching-capability* [switching-capability
   encoding]
            /* Client layer ID */
            |  |     +--rw switching-capability    identityref
            |  |     +--rw encoding                identityref
            /* Adaptation bandwidth available for the client layer */
            |  |     +--rw bandwidth?              te-bandwidth
```

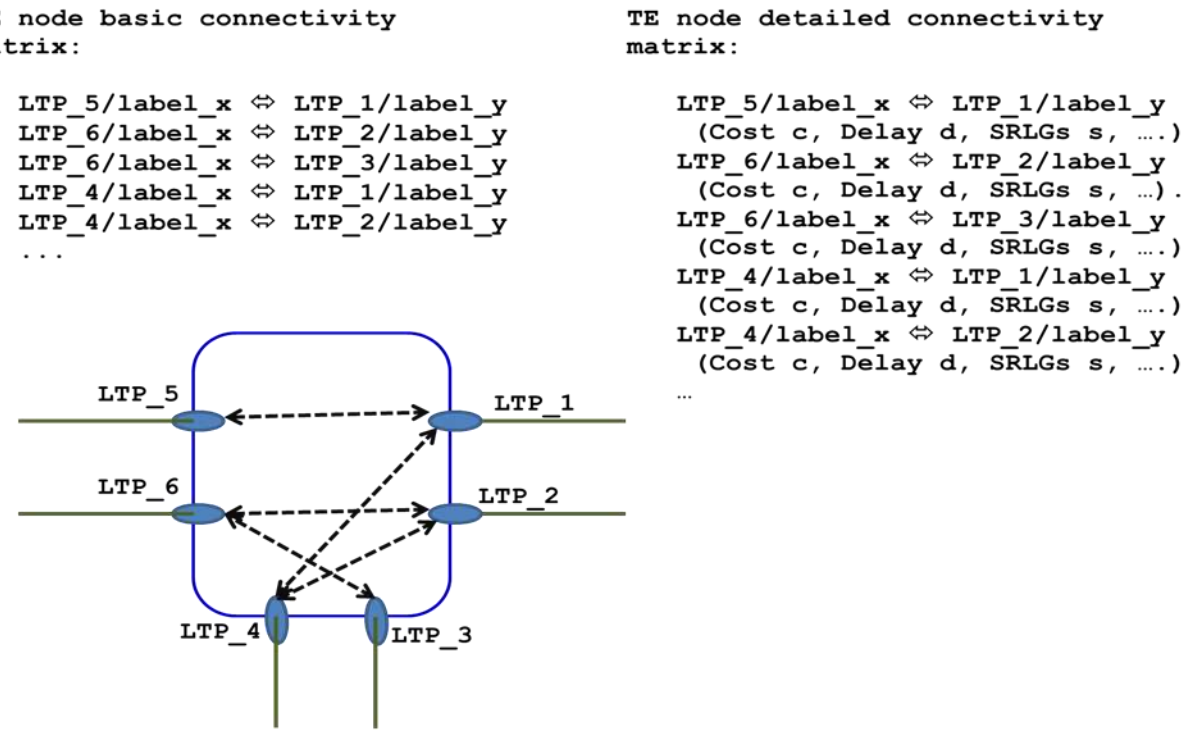

Figure 4. TE Node Connectivity Matrix

- **TE node basic connectivity matrix** - a TE node attribute describing the TE node's switching capabilities/limitations in the form of permissible switching combinations of the TE node's LTP/label pairs (see Figure 4, upper left). From the point of view of a potential TE path arriving at the TE node at a given inbound LTP/label, the node's basic connectivity matrix describes permissible outbound LTP/label pairs for the TE path to leave the TE node.

- **TE node detailed connectivity matrix** - TE node basic connectivity matrix extended to provide a set of costs (such as intra-node summary TE metric, delay, SRLGs, etc.) associated with each connectivity matrix entry (see Figure 4, upper right).

```
/* TE node connectivity matrix */
          |  +--rw connectivity-matrix* [id]
          |     +--rw id                          uint32
          |     +--rw from  /* left LTP */
```

```
               |     |  +--rw tp-ref?   leafref
               |     +--rw to    /* right LTP */
               |     |  +--rw tp-ref?   leafref
               |     +--rw is-allowed?                boolean

              /* Connectivity matrix entry label range */
               |     +--rw label-restriction* [inclusive-exclusive
     label-start]
               |     |  +--rw inclusive-exclusive    enumeration
               |     |  +--rw label-start            rt-
     types:generalized-label
               |     |  +--rw label-end?             rt-
     types:generalized-label
               |     |  +--rw range-bitmap?          binary

             /* Connectivity matrix entry underlay TE path(s) */
               |     +--rw underlay! {te-topology-hierarchy}?
               |     |  +--rw primary-path
               |     |  +--rw backup-path* [index]
              /* Connectivity matrix entry protection type */
               |     |  +--rw protection-type?   uint16
              /* Connectivity matrix entry supporting TE tunnels */
               |     |  +--rw tunnels
              /* Connectivity matrix entry bandwidth parameters */
               |     +--rw max-lsp-bandwidth* [priority]

     ..........................................................................
              /* Connectivity matrix entry metrics (vector of costs) */
               |     +--rw te-default-metric?         uint32
               |     +--rw te-delay-metric?           uint32
               |     +--rw te-srlgs
               |     |  +--rw value*   te-types:srlg
               |     +--rw te-nsrlgs {nsrlg}?

     ..........................................................................
              /* Connectivity matrix entry ID */
               |     |  +--rw id*   uint32
```

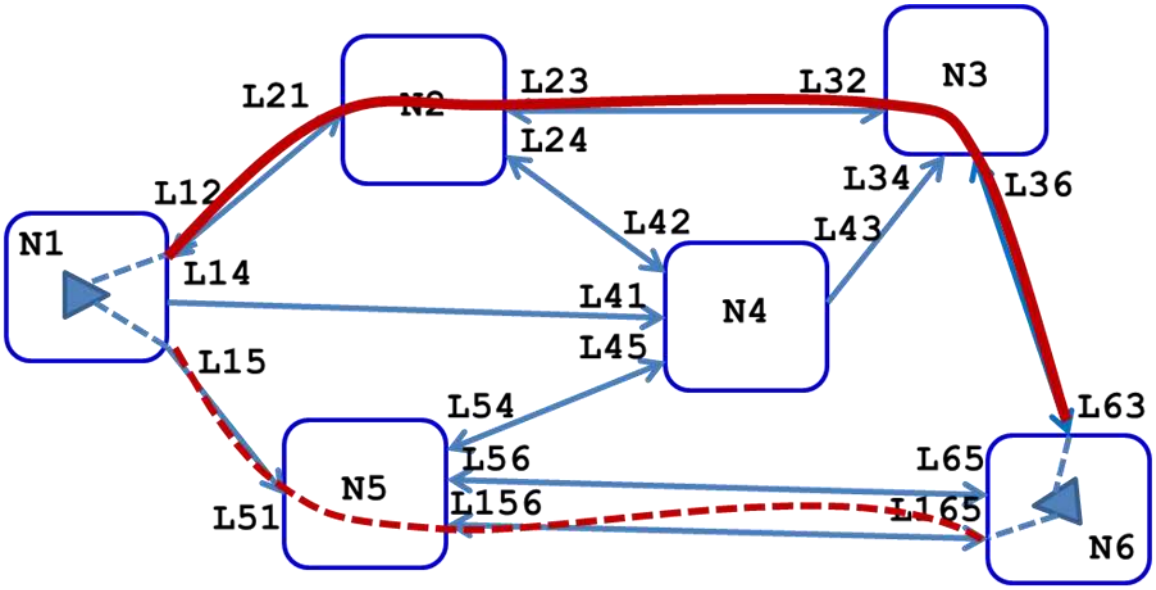

Figure 5. TE Path

- **TE path** - an ordered list of TE node/link IDs (each possibly augmented with labels) that interconnects over a TE topology a pair of TTPs and could be used by a connection (see Figure 5). A TE path could, for example, be a product of a successful path computation performed for a given TE tunnel

---

```
   /* TE path */

         /* TE topology the path is defined in */
    |     |  |  +--rw network-ref?    leafref
         /* Path type (IRO, XRO, ERO, RRO) */
    |     |  |  +--rw path-type?    identityref

         /* TE path elements */
    |     |  |  +--rw path-element* [path-element-id]
    |     |  |     +--rw path-element-id    uint32
    |     |  |     +--rw index?             uint32
    |     |  |     +--rw (type)?
          /* Numbered TE link path element */
    |     |  |        +--:(ip-address)
    |     |  |        |  +--rw ip-address-hop
    |     |  |        |     +--rw address?    inet:ip-address
```

```
|      |  |         |     +--rw hop-type?   te-hop-type
       /* AS number path element */
|      |  |        +--:(as-number)
|      |  |         |  +--rw as-number-hop
|      |  |         |     +--rw as-number?   binary
|      |  |         |     +--rw hop-type?    te-hop-type
       /* Unnumbered TE link path element */
|      |  |        +--:(unnumbered-link)
|      |  |         |  +--rw unnumbered-hop
|      |  |         |     +--rw te-node-id?      inet:ip-address
|      |  |         |     +--rw tp-id?   uint32
|      |  |         |     +--rw hop-type?       te-hop-type
       /* Label path element */
|      |  |        +--:(label)
|      |  |         |  +--rw label-hop
|      |  |         |     +--rw value?   rt-types:generalized-label
|      |  |         |     +--rw direction?       boolean
|      |  |        +--:(sid)
|      |  |           +--rw sid-hop
|      |  |              +--rw sid?   rt-types:generalized-label
```

- **TE path segment** - a contiguous fragment of a TE path

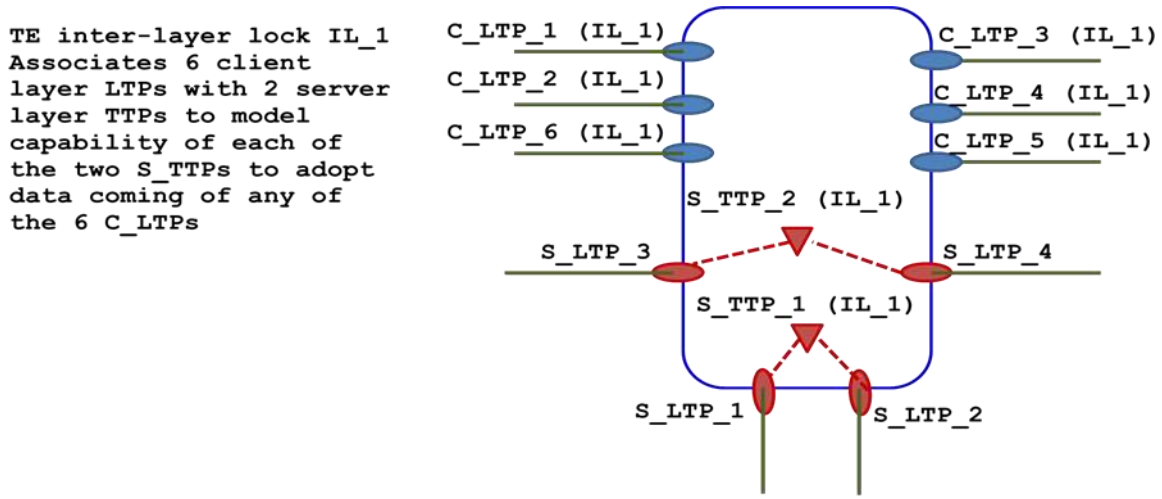

Figure 6. TE Inter-Layer Lock

- **TE inter-layer lock** - a modeling concept describing client-server layer adaptation relationships important for multi-layer traffic engineering. It is an association of M client layer LTPs and N server layer TTPs, within which data arriving at any of the client layer LTPs could be adopted onto any of the server layer TTPs. A TE inter-layer lock is identified by inter-layer lock ID, which is unique across all TE topologies provided by the same provider. The client layer LTPs and the server layer TTPs associated by a given TE inter-layer lock share the same inter-layer lock ID value.

  In Figure 6 a TE inter-layer lock IL_1 associates six client layer LTPs (C_LTP_1 – C_LTP_6) with two server layer TTPs (S_TTP_1 and S_TTP_2). As mentioned, they all have the same attribute –inter-layer lock ID: IL_1, which is the only parameter/value indicating the association. A given LTP may have zero, one or more inter-layer lock IDs. In the case of multiple inter-layer lock IDs, this implies that the data arriving at the LTP can be adopted onto any of TTPs associated with all specified inter-layer locks. For example, C_LTP_1 may be attributed with two inter-layer locks- IL_1 and IL_2. This would mean that C_LTP_1 for adaptation purposes can use not just TTPs associated with inter-layer lock IL_1 (i.e. S_TTP_1 and S_TTP_2 in the Figure), but any of TTPs associated with inter-layer lock IL_2. Likewise, a given TTP may have one or more inter-layer locks, meaning that it can offer the adaptation service to any client layer LTP having an inter-layer lock matching one of its own.

  LTPs and TTPs associated within the same TE inter-layer lock may be hosted by the same (hybrid, multi-layer) TE node or by multiple TE nodes defined in the same or separate TE topologies. The latter case is especially important because TE topologies of different layer networks could be modeled by separate augmentations of the basic (common to all layers) TE topology model.

```
        |  +--rw inter-layer-lock-id?         uint32
```

- **Transitional link** – an alterative method of modeling client-server adaptation relationship. Transitional link is a bi-directional link connecting an LTP in a client layer to an LTP in a server layer, which is associated (via TTP's LLCL) with a server layer TTP capable of adopting of the client layer data onto a TE tunnel terminated by the TTP. Important attributes pf a transitional link are loca;/remote LTP IDs, TE metric and available adaptation bandwidth.

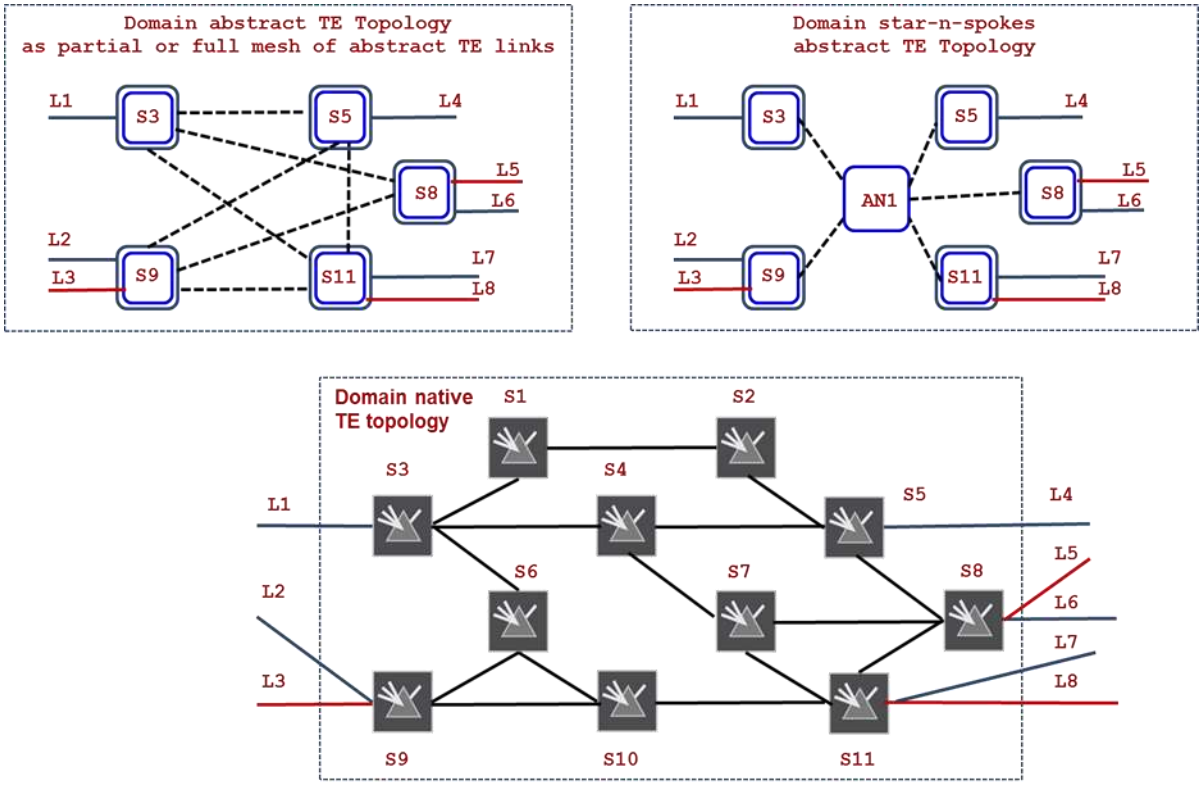

Figure 7. Native and Abstract TE Topologies

- **Native TE topology** - a TE topology as it is known (to full extent and unmodified) to the TE topology provider (see lower part of Figure 7.). A native TE topology might be discovered via various routing protocols and/or subscribe/publish techniques. For example, a first-level TE topology provider (such as a T-SDN Domain Controller, DC) may auto-discover its native TE topology(ies) by participating in the domain's OSPF-TE protocol instance; while a second-level TE topology provider (such as a Hierarchical T-SDN Controller. HC) normally builds its native TE topology(ies) based on TE topologies exposed by each of the subordinate, first- level TE topology providers.

- **Underlay TE topology** - a TE topology that serves as a base for constructing overlay TE topologies.

- **Overlay TE topology** – a TE topology constructed based on one or more underlay TE topologies. Each TE node of the overlay TE topology represents a separate underlay TE topology (that could be mapped onto an arbitrary segment of a native TE topology). Each TE link of the overlay TE topology represents, generally speaking, an arbitrary TE path in one of the underlay TE topologies. The overlay TE topology and the supporting underlay TE topologies may represent separate layer networks (e.g. OTN/ODUk and WDM/OCh respectively) or the same layer network.

- **Abstract TE topology** – an overlay TE topology created by a provider to describe its network in some abstract way. An abstract TE topology contains at least one abstract TE topology element, such as TE node or TE link. An abstract TE topology is built based on contents of one or more of the provider's native TE topologies (serving as underlay(s)), the provider's policies and the client's preferences (see upper part of Figure 7).

- **Customized TE topology** – a TE topology tailored for a given provider's client. A customized TE topology is usually but not always an abstract TE topology. For example, a given abstract TE topology could be exposed to a group or all provider's clients (in which case the abstract TE topology is not a customized TE topology). Likewise, a given naive TE topology could be customized for a given client (for example, by removing high delay TE links the client does not care about). So customized TE topology is not an abstract TE topology, because it does not contain abstract TE topology elements

- **TE inter-domain plug** – a TE link attribute meaningful for open-ended inter-domain/access TE links. It contains a network-wide unique value (inter-domain plug ID) that identifies in the network a connectivity supporting the inter-domain/access TE link in question. It is expected that a given pair of neighboring domain TE topologies (provided by separate providers) will have each at least one open-ended inter-domain/access TE link with a TE inter-domain plug matching to one provided by its neighbor, thus allowing for a client of both domains to identify adjacent nodes in the separate neighboring TE topologies and resolve the open-ended inter-domain/access TE links by connecting them regardless of the links respective local/remote node ID/link ID attributes. Inter-domain plug IDs may be assigned and managed by a central network authority. Alternatively, inter-domain plug IDs could be dynamically auto-discovered (e.g. via LMP protocol).

```
        +--rw external-domain
           |  +--rw network-ref?             leafref
           |  +--rw remote-te-node-id?       te-types:te-node-id
           |  +--rw remote-te-link-tp-id?    te-types:te-tp-id
           |  +--rw plug-id?                 uint32
```

1.3. Abstract TE Topology Calculation, Configuration and Maintenance

The TE Topology Model does not prescribe what and how abstract TE topologies are computed, configured, manipulated and supported by a TE network (e.g. transport network) provider. However, it is assumed that:

- All TE topologies, native or abstract, conveyed to the same or different clients, are largely independent one from another. This implies that each TE topology, generally speaking, has an independent name space for TE node and link IDs, SRLGs, etc. (possibly overlapping with the name spaces of other TE topologies);

- All abstract TE topologies are bound to the respective underlay native or abstract TE topologies only by the overlay/underlay relationships defined by the TE Topology Model, but, otherwise, the abstract TE topologies are decoupled from their respective underlay TE topologies.

It is envisioned that an original set of abstract TE topologies is produced by a TE network provider for each of its clients based on the provider's local configurations and/or policies, as well as the client-specific profiles. The original set of abstract TE topologies offered to a client may be accepted by the client as-is. Alternatively, the client may choose to negotiate/re-configure the abstract TE topologies, so that the latter optimally satisfy the client's needs. In particular, for each of the abstract TE topologies the client may request adding/removing TE nodes, TE links, TTPs and/or modifying re-configurable parameters of the existing components. The client may also request different optimization criteria as compared to those used for the original abstract TE topology optimization, or/and specify various topology-level constraints. The provider may accept or reject all or some abstract TE topology re-configuration requests. Hence, the abstract TE topology negotiation process may take multiple iterations before the provider and each of its clients agree upon a set of abstract TE

topologies and their contents. Furthermore, the negotiation process could be repeated over time to produce new abstract TE topologies optimal to best suit evolving circumstances.

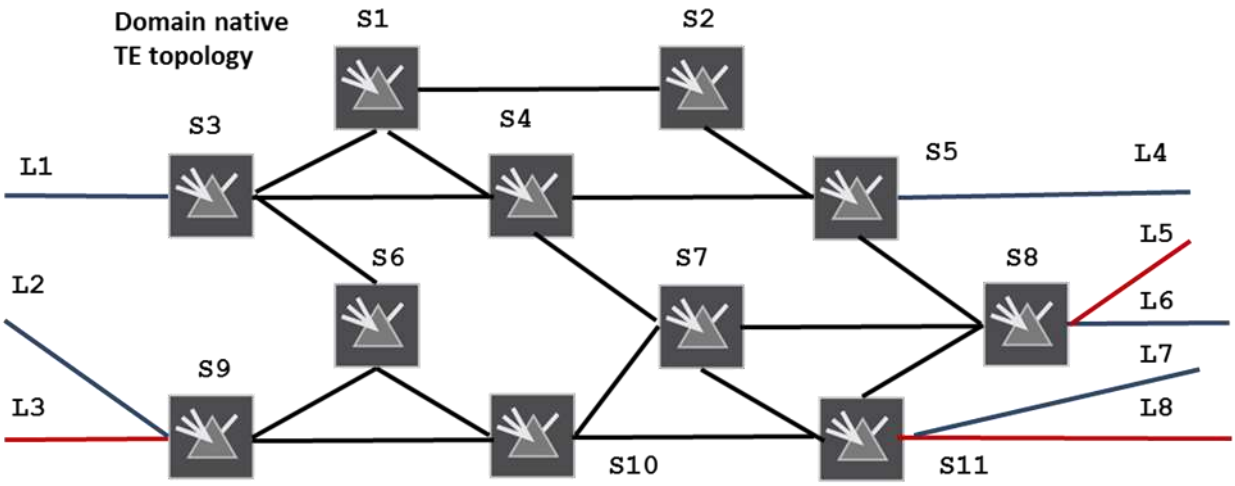

Figure 8. Native Transport Network Domain TE Topology as an Underlay for Abstract TE Topologies

Let's assume that a native transport network domain TE topology to be as depicted in Figure 8. The popular types of abstract TE topologies based on this native TE topology as an underlay are described in the following sections.

1.3.1. Single-Node Abstract TE Topology

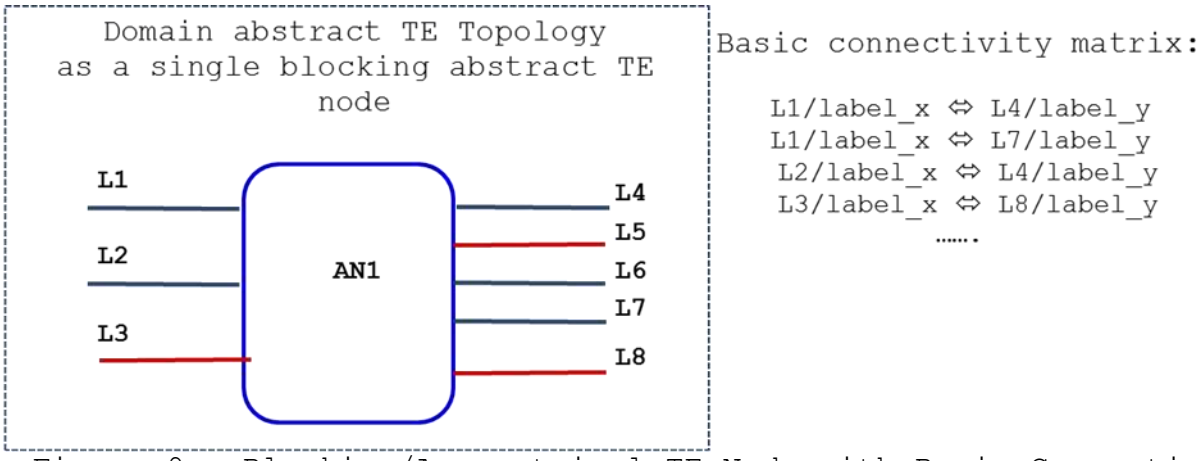

Figure 9. Blocking/Asymmetrical TE Node with Basic Connectivity Matrix Attribute

In Figure 9, the transport network domain is presented to a client as a one-node abstract TE topology, where the single TE node (AN1) represents the entire domain and terminates all of the inter-domain/access TE links connecting the domain to its adjacent domains (i.e. TE links L1...L8). Because AN1 represents the entire domain the node's Underlay TE Topology attribute matches the ID of one of the domain's native TE topologies (e.g. one presented in Figure 8). [Note: all or some of the underlay TE topologies a given abstract TE topology depends on could be catered to the client by the provider along with the abstract TE topology in question or upon separate request(s) issued by the client.]

One important caveat about abstract TE node AN1 is that it should be considered as an asymmetrical/blocking switch, because, generally speaking, it is not guaranteed that a suitable TE path exists between any given pair of inter-domain TE links into/out of the domain. This means from the TE Topology model point of view that there are certain limitations as to how AN1's LTPs could be interconnected inside/across the TE node. The model allows for asymmetrical/blocking switches by specifying for the associated TE nodes a non-empty basic connectivity matrix attribute describing permissible inbound-outbound TE link/label switching combinations. It is assumed that the provider's path computer can compute a set of optimal TE paths, connecting inbound TE link/label_x <=> outbound TE link/label_y combinations inside the abstract TE node over the TE node's underlay TE topology. Based on the results of such computations, AN1's

connectivity matrix can be (re-)generated and (re-)conveyed to the abstract TE topology client.

A richer version of the basic connectivity matrix is the detailed connectivity matrix. The latter not only describes permissible inbound TE link/label_x <=> TE link/label TE link/label_y switching combinations, but also provides connectivity matrix entry specific vectors of various costs/metrics (in terms of delay, bandwidth, intra-node SRLGs and summary TE metrics) that a potential TE path will accrue, should a given connectivity matrix entry be selected by the path for crossing the TE node (see Figure 10).

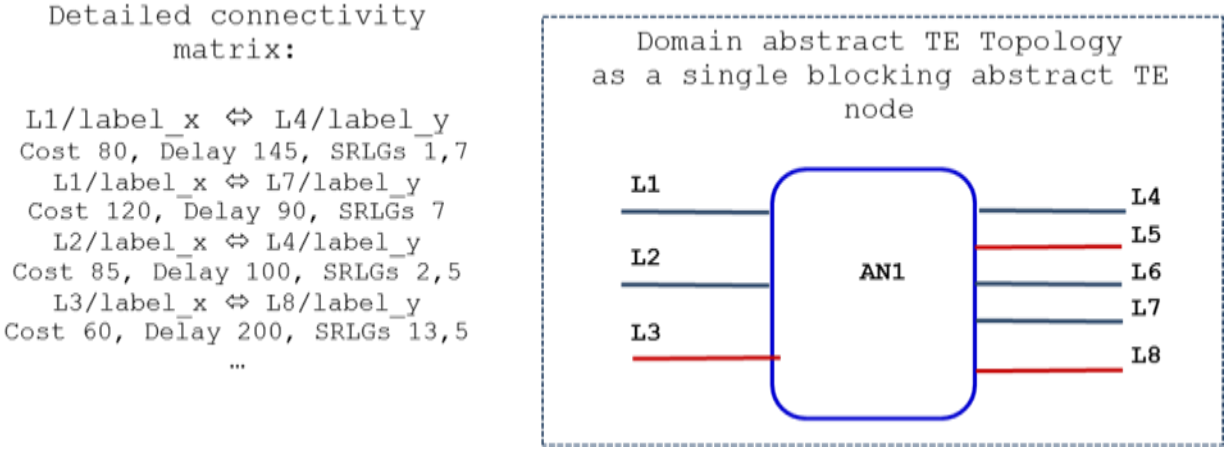

Figure 10. Blocking/Asymmetrical TE Node with Detailed Connectivity Matrix Attribute

1.3.2. Full Mesh Link Abstract TE Topology

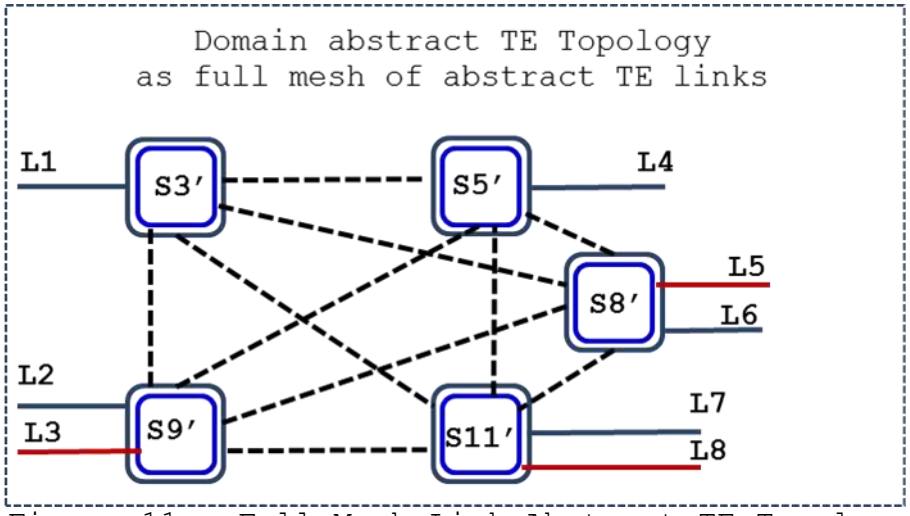

Figure 11. Full Mesh Link Abstract TE Topology

In Figure 11, the transport network domain is abstracted in the following way.

- Each of the underlay native TE topology border TE nodes (i.e., the TE nodes terminating at least one inter-domain/access TE link, such as TE nodes S3 or S11 in Figure 8) is represented in the abstract TE topology as a separate abstract TE node, matching one-for-one to the respective border TE node of the underlay TE topology. For example, S3' of the abstract TE topology represents S3 of the underlay TE topology in Figure 8. [Note that such a relationship is modeled via Supporting Node attribute of TE node S3' specifying the ID of S3, as well as the ID of the TE topology where S3 is defined (i.e. TE topology in Figure 8)]. Likewise, S9' represents S9, S11' represents S11 and so forth;

- TE nodes S3', S5', S8', S9' and S11' are interconnected via a full mesh of abstract TE links. It is assumed that the provider's path computer can compute a set of optimal TE paths over one or more of underlay TE topologies (such as presented in Figure 8)- one for each of said abstract TE links; and the provider can set up the TE tunnels in the network supporting each of the abstract TE links, either during the abstract TE topology configuration (in the case of committed/pre-established abstract TE links), or at the time the first client's connection is placed on the abstract TE link in question (the case of uncommitted abstract TE links). [Note that so (re-)computed TE paths, as well as the IDs of respective underlay TE topologies used for their computation are normally catered to the client in the Underlay TE path attribute of the associated abstract TE links]

The configuration parameters of each of the abstract TE links (such as layer ID, bandwidth and protection requirements, preferred TE paths across the underlay TE topology for the primary and backup connections, etc.) are expected to be found in the abstract TE topology profiles/templates locally configured with the provider or pushed to the provider by the client via the policy NBI. Each of the abstract TE links may be later re-configured or removed by direct configuration requests issued by the client via TE Topology NBI. Likewise, additional abstract TE links may be requested by the client at any time.

Some possible variants/flavors of the Full Mesh Link Abstract TE Topology described above are:

- Partial Mesh Link Abstract TE Topology (where some of the abstract TE links from the full mesh are missing);

- Double Mesh Link Abstract TE Topology (where each pair of abstract TE nodes is connected via two diverse abstract TE links).

1.3.3. Star-n-Spokes Abstract TE Topology

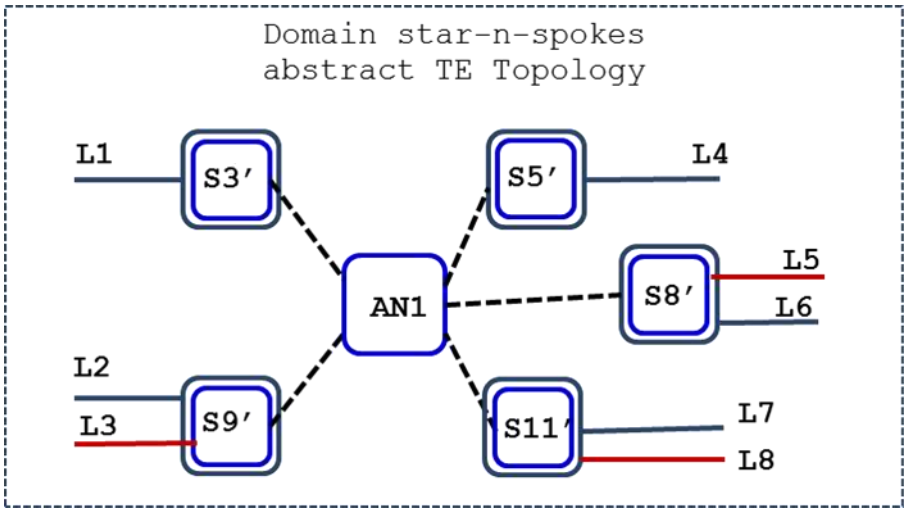

Figure 12. Star-n-Spoke Abstract TE Topology

The Full Mesh Link Abstract TE Topology suffers from the n-squared problem; that is, the number of required abstract TE links is proportional to square of the number of native TE topology border TE nodes. This problem can be mitigated (i.e., the number of required abstract TE links may be significantly reduced) by adding, to the abstract TE topology, an additional abstract TE node (the star) representing one or several interconnected non-border TE nodes from the native TE topology. Abstract TE links in the Star-n-Spokes Topology connect the star with all other TE nodes of the topology (the spokes). For example, abstract TE node AN1 in Figure 12 could represent collectively TE nodes S7, S10 and S4 of the native TE topology (see Figure 8) with abstract TE links connecting AN1 with all other TE nodes in the Star-n-Spokes Abstract TE Topology in Figure 12.

In order to introduce a composite abstract TE node, (e.g. AN1 in Figure 12) representing in a given abstract TE topology an arbitrary segment of another TE topology (e.g. TE nodes S7, S12 and S4 of the TE topology in Figure 8) the TE topology provider is expected to perform the following operations:

- Copy the TE topology segment to be represented by the abstract TE node (i.e. TE nodes S7, S10 and S4 in Figure 8, as well as the TE links interconnecting them) into a separate auxiliary TE topology (with a separate TE topology ID);

- Set for each TE node and TE link of the auxiliary TE topology the Supporting Node/Link attribute matching the original TE topology ID, as well as the ID of the respective original TE node/link of the original TE topology. For example, if S7" of the auxiliary TE topology is a copy of S7 of the original TE topology, the Supporting Node attribute of S7" will specify the ID of the original TE topology (presented in figure 8) and the ID of S7;

- Set for the abstract TE node AN1 the Underlay TE Topology attribute matching the auxiliary TE Topology ID

Furthermore, the Star-n-Spokes Abstract TE topology provider is expected to:

- Compute/provision TE paths/tunnels supporting each of the abstract TE links in Figure 12 (i.e. abstract TE links connecting the spokes to the star, AN1) as described in 1.3.2;

- Generate the AN1's Basic/Detailed Connectivity Matrix attribute based on intra-node path computations performed on the AN1's underlay (i.e. auxiliary) TE topology and describing permissible inbound TE link/label_x outbound TE link/label_y switching combinations as described in 1.3.1

1.3.4. Arbitrary Abstract TE Topology

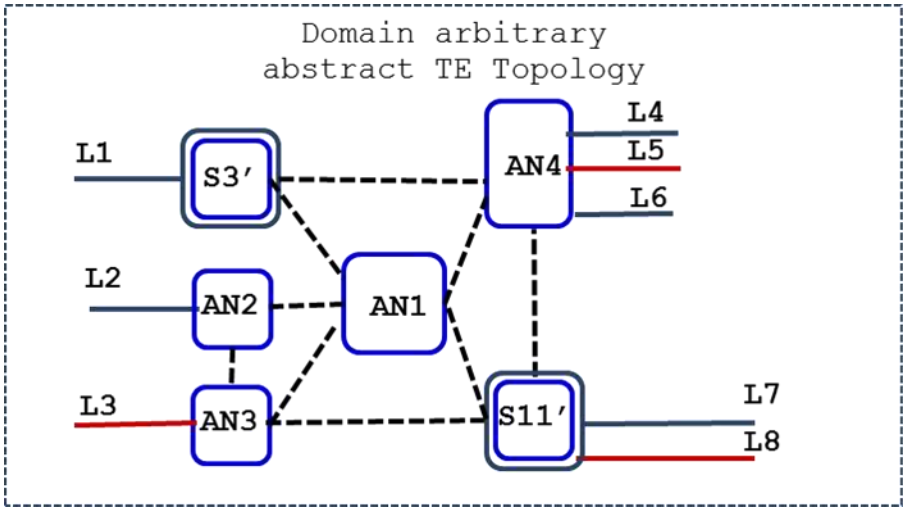

Figure 13. Arbitrary Abstract TE Topology

To achieve an optimal tradeoff between the number of components, the amount of information exposed by a transport network provider and the

amount of path computations required to keep said information up-to-date, the provider may present the TE network domain as an arbitrary abstract TE topology comprised of any number of abstract TE nodes interconnected by abstract TE links (see Figure 13). Each of the abstract TE nodes can represent a single or several interconnected TE nodes from the domain's underlay (native or lower level abstract) TE topology, or a fraction of an underlay TE node. [Note that each of the abstract TE nodes of the TE topology in Figure 13 is expected to be introduced and maintained by the provider following the instructions as described in 1.3.3; likewise, each of the abstract TE links of the topology is expected to be computed, provisioned and maintained as described in 1.3.2]

1.3.5. Customized Abstract TE Topologies

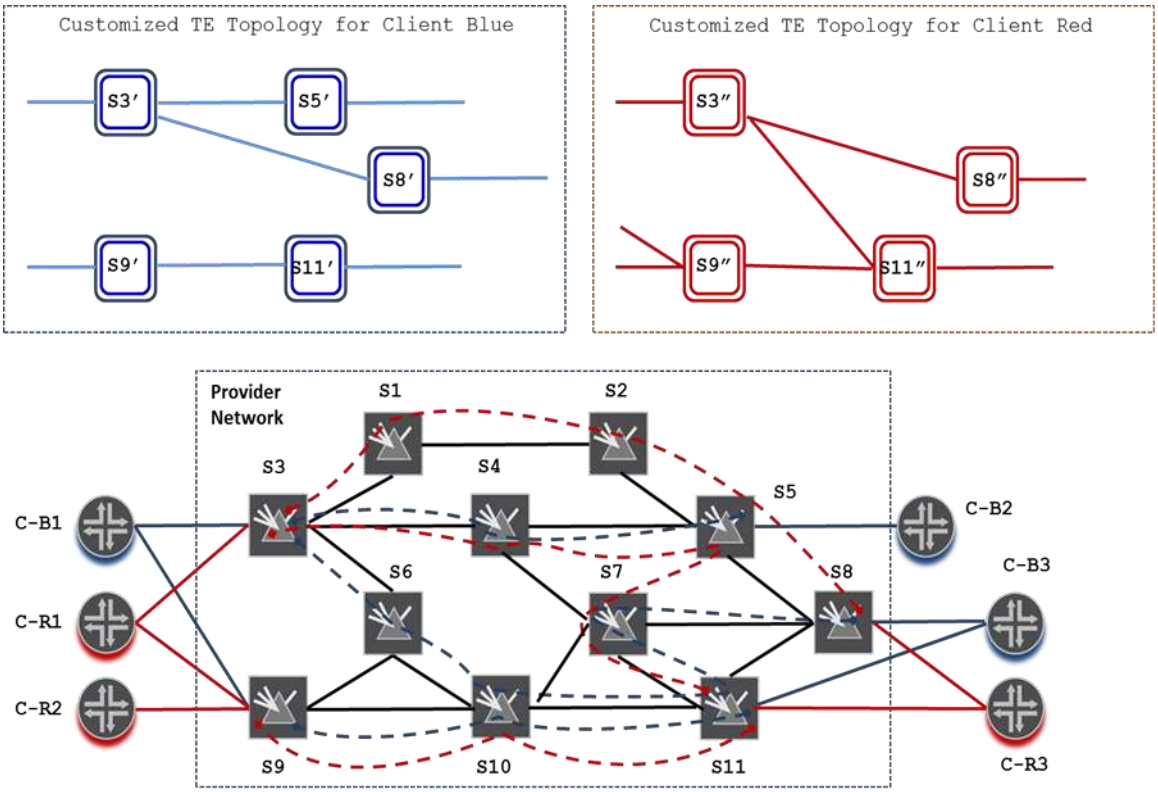

Figure 14. Customized Abstract TE Topology(ies)

A transport network/domain provider may serve more than one client. In such a case, the provider "slices" the network/domain resources and exposes a slice for each of the clients in the form of a customized abstract TE topology. In Figure 14, the provider serves

two clients (Blue and Red). Client Blue is provided with the Blue abstract TE topology supported by the blue TE tunnels or paths in the underlay (native) TE topology (depicted in the Figure with blue broken lines). Likewise, client Red is provided with the Red abstract TE topology supported by the red TE tunnels or paths in the underlay TE topology.

1.3.6. Hierarchical Abstract TE Topologies

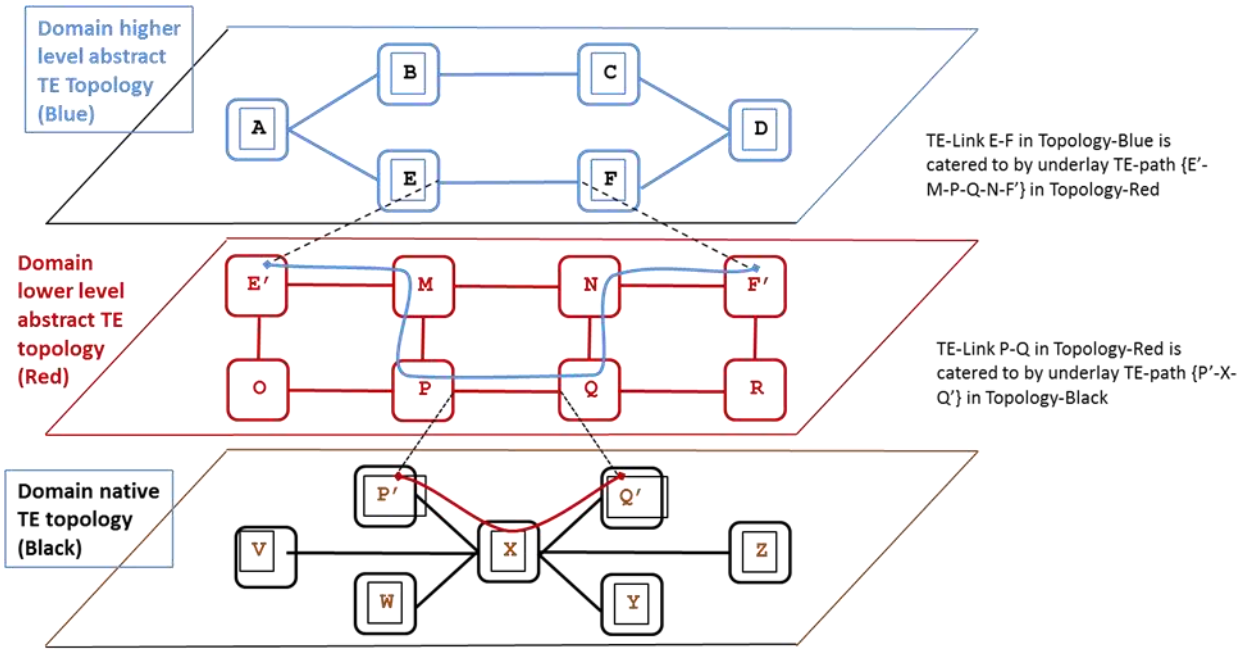

Figure 15. Hierarchy of Abstract TE Topologies

As previously mentioned, an underlay TE topology for a given abstract TE topology component does not have to be one of the domain's native TE topologies – another (lower level) domain's abstract TTE topology can be used instead. This means that abstract TE topologies are hierarchical in nature.

Figure 15 provides an example of abstract TE topology hierarchy. In this Figure the blue topology is a top level abstract TE topology catered to by the provider to one of the domain's clients. One of the TE links of the blue topology – link EF – is supported by a TE path E'-M-P-Q-N-F' computed in the underlay TE topology (red topology), which happens to be domain's (lower level) abstract TE topology.. Furthermore, as shown, the TE link PQ – one of the TE links comprising the E'-M-P-Q-N-F' path – is supported by its own underlay

TE path, P'-X-Q' - computed on one of the domain's native TE topologies.

Importantly, each TE link and TE node of a given abstract TE topology has, generally speaking, its individual stack/hierarchy of underlay TE topologies.

1.4. Merging TE Topologies Provided By Multiple Providers

A client may receive TE topologies provided by multiple providers, each of which managing a separate domain of an interconnected multi-domain transport network. In order to make use of said topologies, the client is expected to merge (inter-connect) the provided TE topologies into one or more client's native TE topologies, each of which homogeneously representing the multi-domain transport network. This makes it possible for the client to select end-to-end TE paths for its TE tunnel connections traversing multiple domains.

In particular, the process of merging TE topologies includes:

- Identifying neighboring TE domains and locking their TE topologies horizontally by connecting their inter-domain open-ended TE links;

- Renaming TE node, link, and SRLG IDs into ones allocated from a separate name space; this is necessary because all TE topologies are considered to be, generally speaking, independent with a possibility of clashes among TE node, link or SRLG IDs. Original TE node/link IDs along with the original TE topology ID are stored in the Source attribute of the respective TE nodes/links of the merged TE topology;

- Locking, TE topologies associated with different layer networks vertically according to provided TE inter-layer locks; this is to facilitate inter-layer path computations across multiple TE topologies provided by the same topology provider.

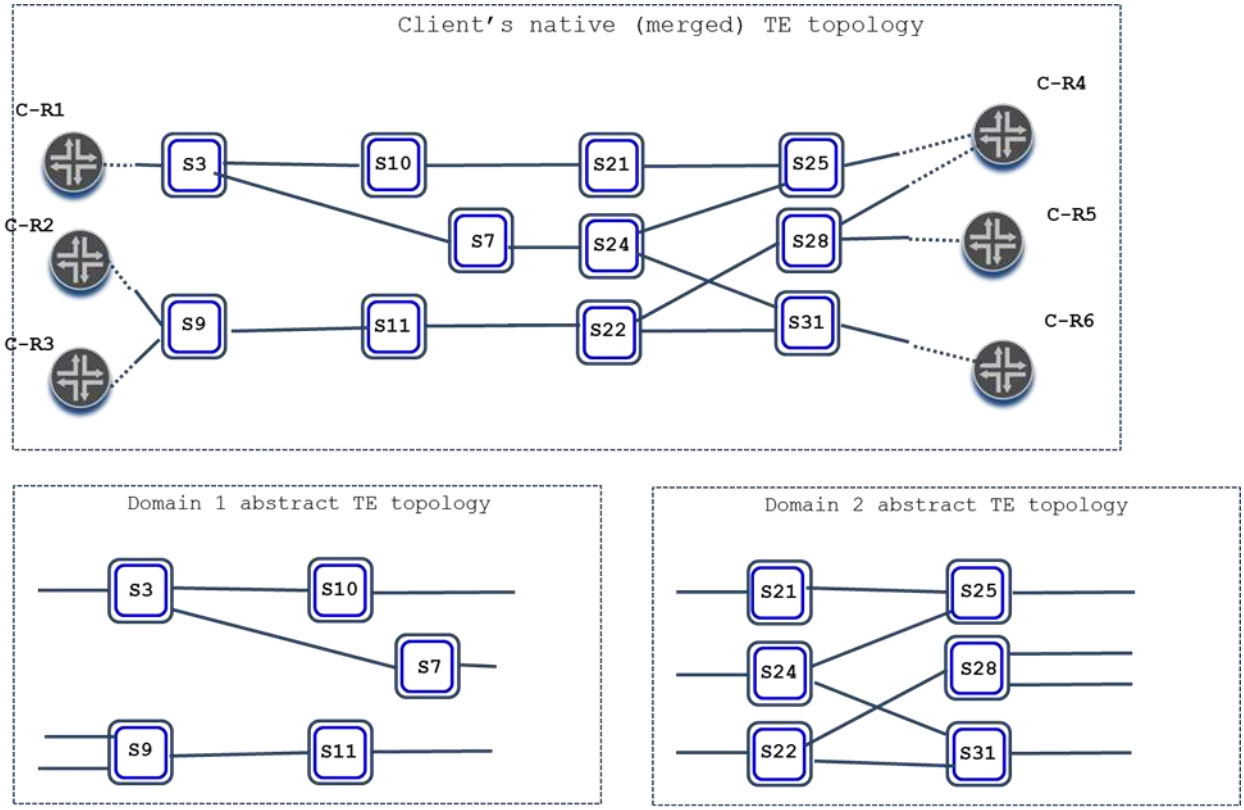

Figure 16. Merging Domain TE Topologies

Figure 16 illustrates the process of merging, by the client, of TE topologies provided by the client's providers.

In the Figure, each of the two providers caters to the client a TE topology (abstract or native), describing the network domain under the respective provider's control. The client, by consulting the attributes of the open-ended inter-domain/access TE links - such as TE inter-domain plugs or remote TE node/link IDs - is able to determine that:

1. the two domains are adjacent and are interconnected via three inter-domain TE links, and;

2. each domain is connected to a separate customer site, connecting the left domain in the Figure to customer devices C-11 and C-12, and the right domain to customer devices C-21, C-22 and C-23.

Therefore, the client interconnects the open-ended TE links, as shown on the upper part of the Figure.

As mentioned, one way to interconnect the open-ended inter-domain/access TE links of neighboring domains is to mandate the providers to specify remote nodeID/linkID attributes in the provided inter-domain/access TE links. This, however, may prove to be not flexible. For example, the providers may not be aware of the respective remote nodeID/linked values. More importantly, this option does not allow for the client to mix-n-match multiple (more than one) TE topologies catered by the same providers (see the next section). Another, more flexible, option to resolve the open-ended inter-domain/access TE links is by decorating them with the TE inter-domain plug attribute. The attribute specifies inter-domain plug ID - a network-wide unique value that identifies on the network connectivity supporting a given inter-domain/access TE link. Instead of specifying remote node ID/link ID, an inter-domain/access TE link may provide a non-zero inert-domain plug ID. It is expected that two neighboring domain TE topologies (provided by separate providers) will have each at least one open-ended inter-domain/access TE link with a TE inter-domain plug matching to one provided by its neighbor. For example, the inter-domain TE link originating from node S5 of the Domain 1 TE topology (Figure 8) and the inter-domain TE link coming from node S3 of Domain2 TE topology may specify matching TE inter-domain plugs (i.e. carrying the same inter-domain plug ID). This would allow for the client to identify adjacent nodes in the separate neighboring TE topologies and resolve the inter-domain/access TE links connecting them regardless of their respective nodeIDs/linkIDs (which, as mentioned, could be allocated from independent name spaces).

Inter-domain plug IDs may be assigned and managed by a central network authority. Alternatively, inter-domain plug IDs could be dynamically auto-discovered (e.g. via LMP protocol).

Furthermore, the client renames the TE nodes, links and SRLGs offered in the abstract TE topologies by assigning to them IDs allocated from a separate name space managed by the client. Such renaming is necessary, because the two abstract TE topologies may have their own name spaces, generally speaking, independent one from another; hence, ID overlaps/clashes are possible. For example, both TE topologies have TE nodes named S7, which, after renaming, appear in the merged TE topology as S17 and S27 respectively. IDs of the original (i.e. abstract TE topology) TE nodes/links along with the ID of the abstract TE topology they belong to are stored in the Source attribute of the respective TE nodes/links of the merged TE topology. For example, the Source attribute of S27 will contain S7 and the TE topology ID of the abstract TE topology describing domain 2.

Once the merging process is complete, the client can use the merged TE topology for path computations across both domains, for example, to compute a TE path connecting C-11 to C-23.

1.4.1. Dealing With Multiple Abstract TE Topologies Provided By The Same Provider

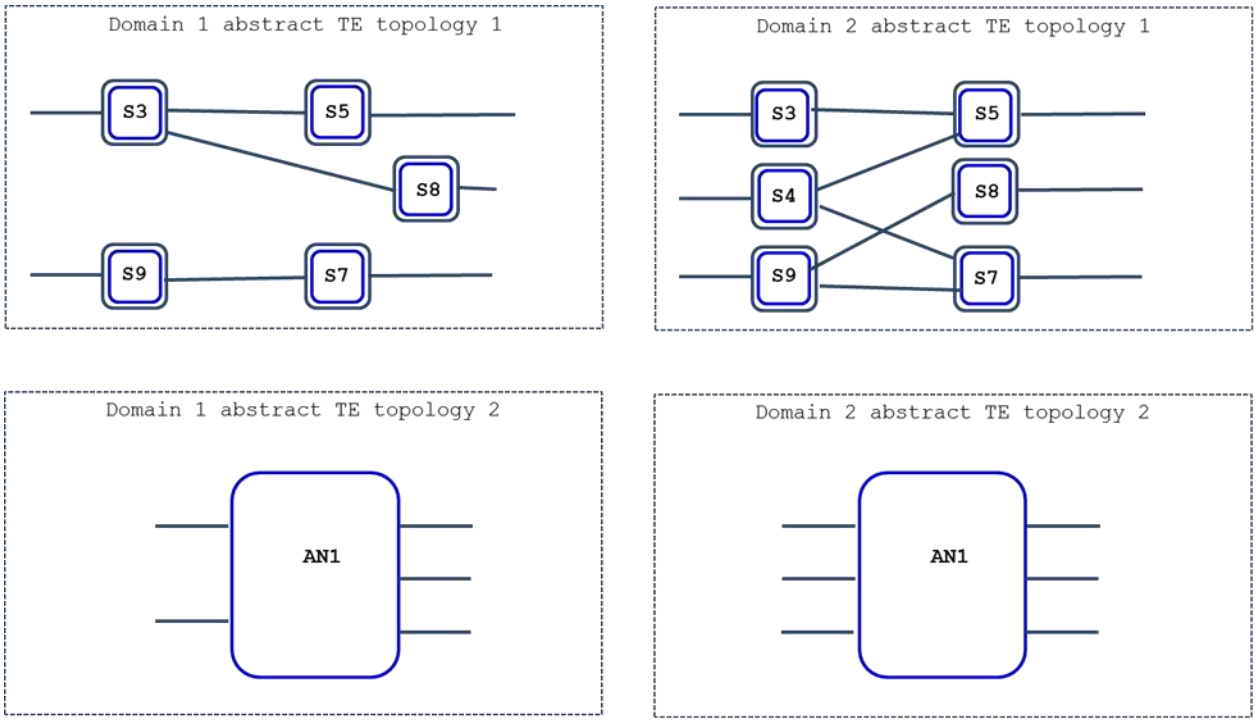

Figure 17. Multiple Abstract TE Topologies Provided By TE Topology Providers

A given provider may expose more than one abstract TE topology to the client. For example, one abstract TE topology could be optimized based on a lowest-cost criterion, while another one could be based on best possible delay metrics, while yet another one could be based on maximum bandwidth availability for the client connections. Furthermore, the client may request all or some providers to expose additional abstract TE topologies, possibly of a different type and/or optimized differently, as compared to already-provided TE topologies. In any case, the client should be prepared for a provider to offer to the client more than one abstract TE topology.

It should be up to the client to decide how to mix-and-match multiple abstract TE topologies provided by each of the providers, as well as how to merge them into the client's native TE topologies. The client also decides how many such merged TE topologies it needs to produce and maintain. For example, in addition to the merged TE topology depicted on the upper part of Figure 16, the client may merge the abstract TE topologies received from the two providers, as shown in Figure 17, into the client's additional native TE topologies, as shown in Figure 18.

[Note: allowing for the client mix-n-matching of multiple TE topologies assumes that TE inter-domain plugs (rather than remote nodeID/linked) option is used for identifying neighboring domains and inter-domain/access TE link resolution.]

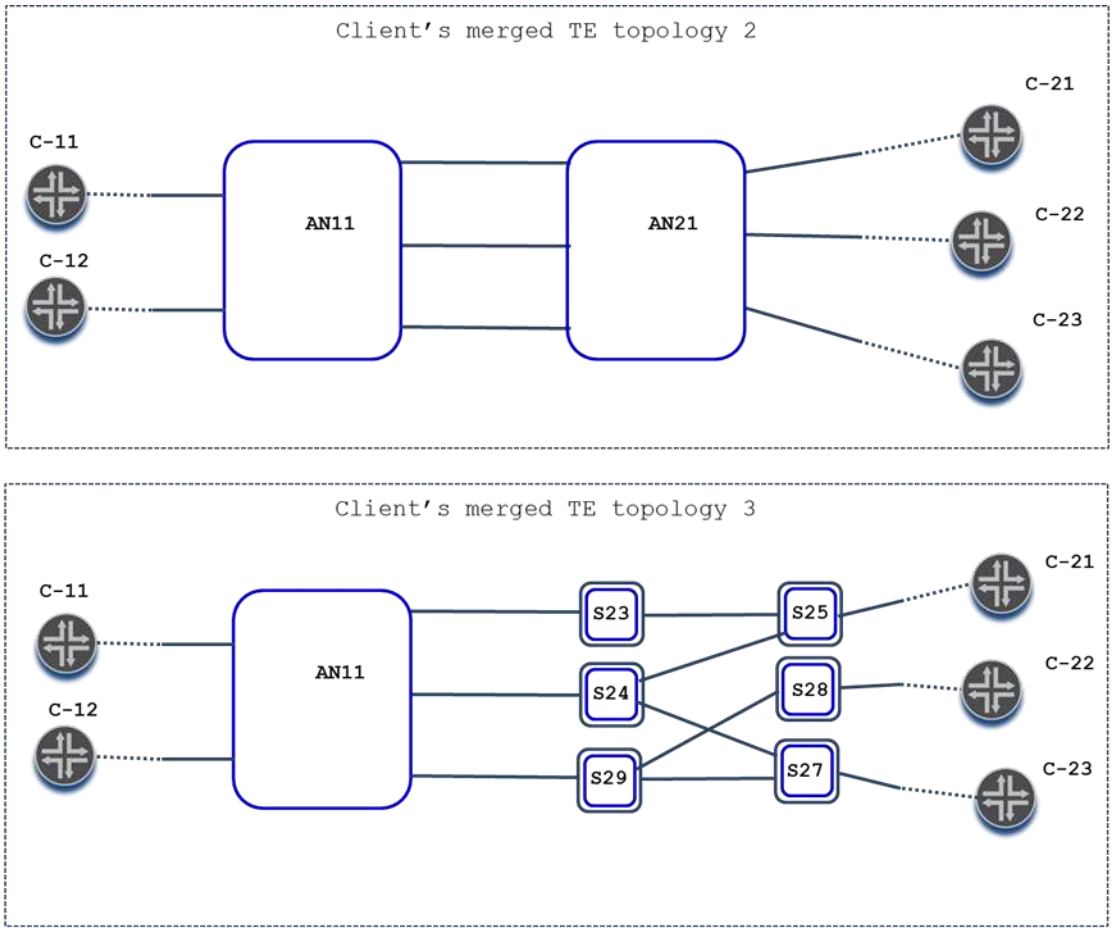

Figure 18. Multiple Native (Merged) Client's TE Topologies

It is important to keep in mind that each of the three native (merged) TE topologies could be used by the client for computing TE paths for any of the multi-domain connections. The choice as to which topology to use for a given connection depends on the connection/tunnel parameters/requirements and the topology's style and optimization criteria.

1.5. Configuring Abstract TE Topologies

When a client receives one or more abstract TE topologies from one of its providers, it may accept the topologies as-is and merge then into one or more of its own native TE topologies. Alternatively, the client may choose to request a re-configuration of one, some or all abstract TE topologies provided by the providers. Specifically, with respect to a given abstract TE topology, some of its TE nodes/links may be requested to be removed, while additional ones may be requested to be added. It is also possible that existing TE nodes/links may be asked to be re-configured. For example, a set of TE links may be requested to be disjoint from each other by configuring the same Non Sharing Risk Link Group (NSRLG) attribute for all links from the set. Such a configuration would force the provider to place TE tunnels supporting the TE links from the set onto sufficiently disjoint TE paths computed in the tunnels underlay TE topology. Furthermore, the topology-wide optimization criteria may be requested to be changed. For example, underlay TE paths supporting the abstract TE links, currently optimized to be shortest (least-cost) paths, may be requested to be re-optimized based on the minimal delay criteria. Additionally, the client may request the providers to configure entirely new abstract TE topologies and/or to remove existing ones. Furthermore, future periodic or one time additions, removals and/or re-configurations of abstract TE topology elements and/or their attributes could be (re-)scheduled by the client ahead of time.

It is the responsibility of the client to implement the logic behind the above-described abstract TE topology negotiation. It is expected that the logic is influenced by the client's local configuration/templates, policies conveyed by client's clients, input from the network planning process, telemetry processor, analytics systems and/or direct human operator commands. Figure 19 exemplifies the abstract TE topology negotiation process. As shown in the Figure, the original abstract TE topology exposed by a provider was requested to be re-configured. Specifically, one of the abstract TE links was asked to be removed, while three new ones were asked to be added to the abstract TE topology.

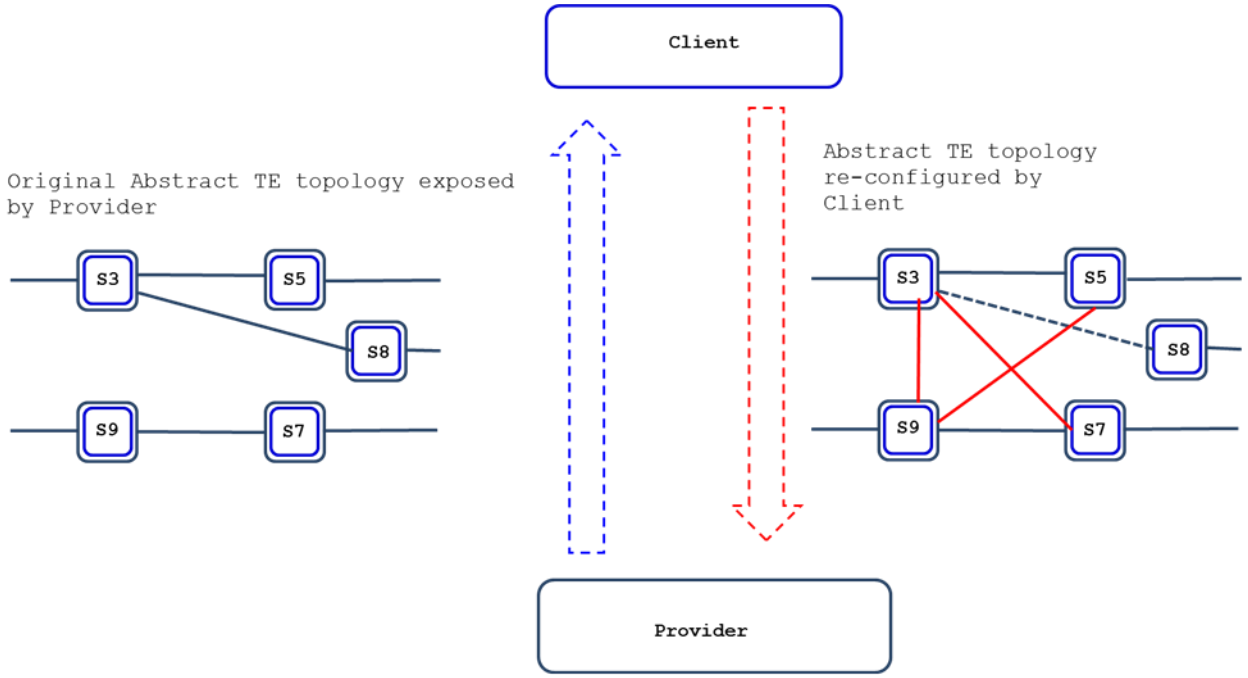

Figure 19. Provider□ Client Abstract TE Topology Negotiation

1.6. TE Tunnel Model

The TE Tunnel Model is written in YANG modeling language. It is defined and developed by the IETF TEAS WG and is documented as "YANG Data Model for Traffic Engineering Tunnels and Interfaces" [I-D.ietf-teas-yang-te]. Among other things the model describes a TE network provider's TE Tunnel data store as it is seen and influenced by a client.

The TE Tunnel Model allows for the provider to convey to each of its clients:

- information on TE tunnels provided to the client that are fully contained within the controlled network domain,
- information on multi-domain TE tunnel segments across the network domain controlled by the provider;
- information on connections/LSPs, supporting TE tunnels and TE tunnel segments;
- updates in response to changes to the client's active TE tunnels/segments and the connections supporting them,

- updates in response to the TE tunnel/segment telemetry/state information the client has expressed an interest in.

The TE Tunnel Model allows for a TE network client to:

- Issue configuration requests to set up, tear down, replace, modify and manipulate end-to-end TE tunnels, as well as segments of multi-domain TE tunnels across the network controlled by the provider;

- Request and obtain information on active TE tunnels/segments and connections supporting them;

- Subscribe to and configure with the provider triggers, pace and contents of the TE tunnel/segment change update notifications;

- Subscribe to and configure with the provider triggers, pace and contents of the TE tunnel/segment event notifications, such as detected alarms, faults, protection/restoration actions, etc..

- Subscribe to and configure with the provider triggers, pace and contents of TE tunnel/segment telemetry (e.g. statistics counters) update notifications.

1.7. TE Tunnel/Transport Service Modeling Constructs

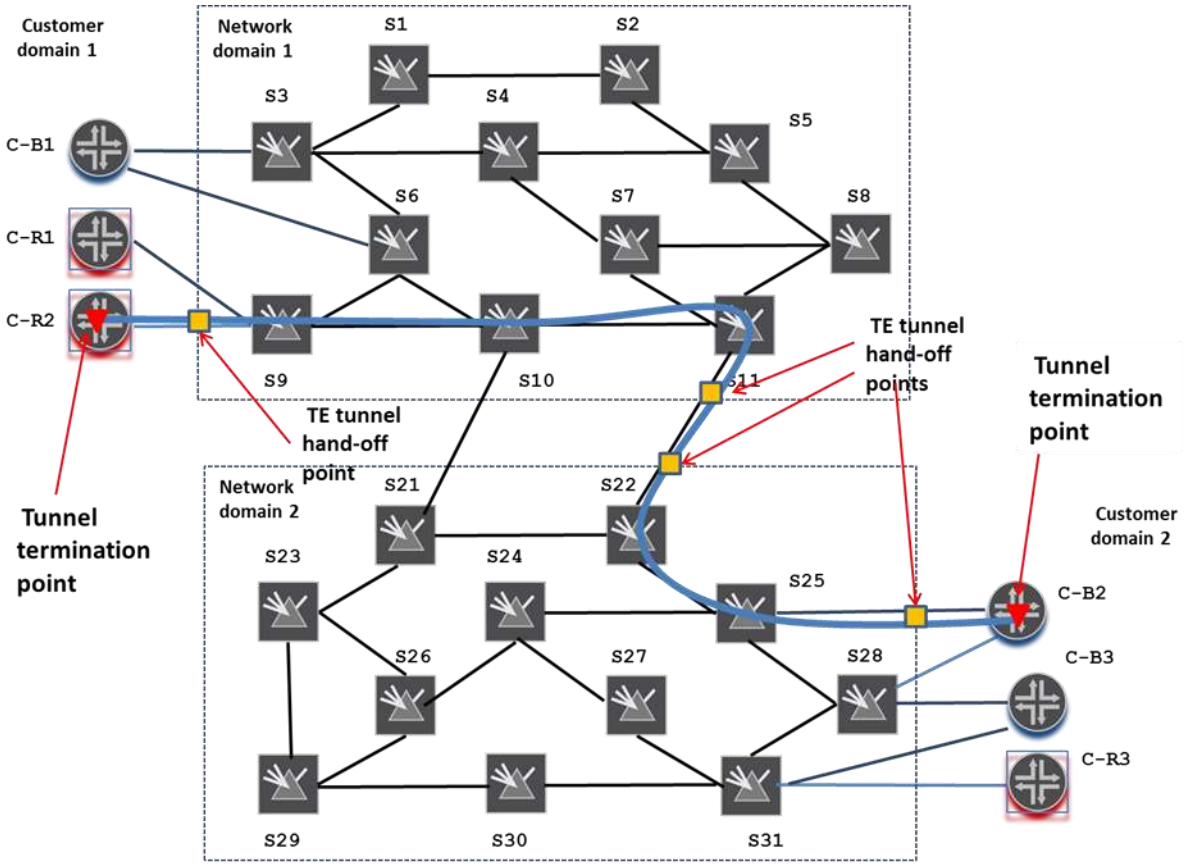

Figure 20. TE tunnel

- **TE tunnel** - a connection-oriented service provided by a layer network of delivery of a client's data between source and destination tunnel termination points. A TE tunnel in a server layer network may support a link in a client layer network (e.g. OCh layer TE tunnel supporting ODU4 link). In Figure 20, a TE tunnel interconnects tunnel termination points resident on switches C-R2 and C-R3. A TE tunnel is realized via (supported by, mapped onto) one or more layer network connections/LSPs

---

```
/* TE tunnel */
 |  +--rw tunnel* [name]
      |  |  +--rw name                    leafref
```

```
      |  |  +--rw identifier?            leafref
   /* TE tunnel configuration parameters */
      |  |  +--rw config
      |  |  |  +--rw name?                   string
      |  |  |  +--rw type?                   identityref
      |  |  |  +--rw identifier?             uint16
      |  |  |  +--rw description?            string
      |  |  |  +--rw switchcap?              identityref
      |  |  |  +--rw encoding?               identityref
      |  |  |  +--rw protection-type?        identityref
      |  |  |  +--rw admin-status?           identityref
      |  |  |  +--rw preference?             uint8
      |  |  |  +--rw reoptimize-timer?       uint16
      |  |  |  +--rw source?                 inet:ip-address
      |  |  |  +--rw destination?            inet:ip-address
      |  |  |  +--rw src-tp-id?              binary
      |  |  |  +--rw dst-tp-id?              binary
      |  |  |  +--rw topology-id?            te-types:te-topology-
   id
      |  |  |  +--rw ignore-overload?        boolean
      |  |  |  +--rw bandwidth-generic?      te-types:te-bandwidth
      |  |  |  +--rw disjointness?           te-types:te-path-
   disjointness
      |  |  |  +--rw setup-priority?         uint8
      |  |  |  +--rw hold-priority?          uint8
      |  |  |  +--rw signaling-type?         identityref
   /* Hierarchy TE tunnel parameters */
      |  |  |  +--rw hierarchical-link-id
      |  |  |  |  +--rw local-te-node-id?      te-types:te-node-id
      |  |  |  |  +--rw local-te-link-tp-id?   te-types:te-tp-id
      |  |  |  |  +--rw remote-te-node-id?     te-types:te-node-id
      |  |  |  |  +--rw te-topology-id?        te-types:te-
   topology-id
   /* Bidirectional TE tunnel parameters */
      |  |  |  +--rw bidirectional
      |  |  |     +--rw association
      |  |  |        +--rw id?              uint16
      |  |  |        +--rw source?          inet:ip-address
      |  |  |        +--rw global-source?   inet:ip-address
      |  |  |        +--rw type?            identityref
      |  |  |        +--rw provisioing?     identityref
   /* TE tunnel state */
      |  |  +--ro state
      |  |  |  +--ro name?                   string
      |  |  |  +--ro type?                   identityref
      |  |  |  +--ro identifier?             uint16
   ...........................................................
```

```
      |  |  |  +--ro oper-status?            identityref
   /* TE tunnel primary path and LSP container */
      |  |  +--rw p2p-primary-paths
      |  |  |  +--rw p2p-primary-path* [name]
      |  |  |     +--rw name
                /* Configuration */
   leafref
      |  |  |     +--rw config
      |  |  |     |  +--rw name?                        string
      |  |  |     |  +--rw preference?                  uint8
      |  |  |     |  +--rw path-setup-protocol?         identityref
      |  |  |     |  +--rw path-computation-method?     identityref
      |  |  |     |  +--rw path-computation-server?     inet:ip-
   address
      |  |  |     |  +--rw compute-only?                empty
      |  |  |     |  +--rw use-cspf?                    boolean
      |  |  |     |  +--rw verbatim?                    empty
      |  |  |     |  +--rw lockdown?                    empty
      |  |  |     |  +--rw named-explicit-path?         leafref
      |  |  |     |  +--rw named-path-constraint?       leafref {te-
   types:named-path-constraints}?
                  /* state */
      |  |  |     +--ro state
      |  |  |     |  +--ro name?                         string
      |  |  |     |  +--ro preference?                   uint8
      |  |  |     |  +--ro path-setup-protocol?          identityref
      |  |  |     |  +--ro path-computation-method?      identityref
      |  |  |     |  +--ro path-computation-server?      inet:ip-
   address
      |  |  |     |  +--ro compute-only?                 empty
      |  |  |     |  +--ro use-cspf?                     boolean
      |  |  |     |  +--ro verbatim?                     empty
      |  |  |     |  +--ro lockdown?                     empty
      |  |  |     |  +--ro named-explicit-path?          leafref
      |  |  |     |  +--ro named-path-constraint?        leafref
   {te-types:named-path-constraints}?
                     /* Computed path */
                     /* Computed path properties/metrics /
      |  |  |     |  +--ro computed-path-properties
      |  |  |     |  |  +--ro path-metric* [metric-type]
      |  |  |     |  |  |  +--ro metric-type            identityref
      |  |  |     |  |  |  +--ro accumulative-value?    uint64
                     /* Computed path affinities */
      |  |  |     |  |  +--ro path-affinities
      |  |  |     |  |  |  +--ro constraints* [usage]
      |  |  |     |  |  |     |  +--ro usage?
   identityref
```

```
          |  |  |     |  |  |     |  +--ro (style)?
          |  |  |     |  |  |     |     +--:(value)
          |  |  |     |  |  |     |     |  +--ro value?            te-
      types:admin-groups
          |  |  |     |  |  |     |     +--:(named)
          |  |  |     |  |  |     |        +--ro affinity-names*
      [name]
          |  |  |     |  |  |     |           +--ro name    string
                     /* Computed path SRLGs */
          |  |  |     |  |  +--ro path-srlgs
          |  |  |     |  |  |  +--ro (style)?
          |  |  |     |  |  |     +--:(values)
          |  |  |     |  |  |     |  +--ro usage?         identityref
          |  |  |     |  |  |     |  +--ro values*        te-
      types:srlg
          |  |  |     |  |  |     +--:(named)
          |  |  |     |  |  |        +--ro constraints* [usage]
          |  |  |     |  |  |           +--ro usage
      identityref
          |  |  |     |  |  |           +--ro constraint
          |  |  |     |  |  |              +--ro srlg-names* [name]
          |  |  |     |  |  |                 +--ro name    string
                     /* Computed path sub-objects */
          |  |  |     |  |  +--ro path-computed-route-objects
      ...............................................................
                     /* LSP (provisioned path) */
          |  |  |     |     +--ro lsp* [source destination tunnel-id
      lsp-id extended-tunnel-id type]
                     /* LSP parameters */
          |  |  |     |        +--ro source                leafref
          |  |  |     |        +--ro destination           leafref
          |  |  |     |        +--ro tunnel-id             leafref
          |  |  |     |        +--ro lsp-id                leafref
          |  |  |     |        +--ro extended-tunnel-id    leafref
          |  |  |     |        +--ro type                  leafref
          |  |  |     |        +--ro signaling-type?       identityref
          |  |  |     +--rw candidate-p2p-secondary-paths
          |  |  |        +--rw candidate-p2p-secondary-path*
      [secondary-path]
          |  |  |           +--rw secondary-path    leafref
          |  |  |           +--rw config
          |  |  |           |  +--rw secondary-path?        leafref
          |  |  |           |  +--rw priority?              uint16
          |  |  |           |  +--rw path-setup-protocol?
      identityref
          |  |  |           +--ro state
          |  |  |              +--ro secondary-path?        leafref
```

```
      |  |  |              +--ro priority?              uint16
      |  |  |              +--ro path-setup-protocol?
identityref
      |  |  |              +--ro active?                boolean

/* TE tunnel secondary path and LSP container */

      |  |  +--rw p2p-secondary-paths
      |  |  |  +--rw p2p-secondary-path* [name]
.....................................................
      |  |  |     +--rw name      leafref
      |  |  |     +--rw config (same as for primary path )
.....................................................
      |  |  |     +--ro state  (same as for primary, except for
disjointedness_state )
   |        |  +--ro disjointness_state?       te-types:te-path-
disjointness.....................................................
      |  |  |        +--ro computed-path-properties (same as for
primary path)
.............................................................
    |  |  |        |  +--ro path-affinities (same as for primary
path)
.............................................................
      |  |  |        |  +--ro path-srlgs    (same as for primary
path)
.............................................................
      |  |  |        |  +--ro path-computed-route-objects
.............................................................
                     /* LSP (provisioned path) */
      |  |  |        +--ro lsp  (same as for the primary LSP)
.....................................................
```

- **Tunnel termination point (TTP)** – a physical device inside a given node/switch realizing a TE tunnel termination function in a given layer network, as well as the TE tunnel's adaptation function provided for client layer network(s). One example of tunnel termination point is an OCh layer transponder. [Note: Tunnel termination points are not to be confused with TE tunnel termination points, which are TE representations of physical tunnel termination points. Similar to physical switches and links of the network, such as depicted in Figure 20, being represented on a TE topology describing the network as TE nodes and TE links, (physical) tunnel termination points (TTPs) are represented as TE tunnel termination points (TE TTPs, see 1.2) hosted by the TE nodes. For example, a provisioned connection/LSP starts on a source TTP, goes through a chain of physical links and stops on a destination TTP. In contrast, TE path (e.g. result of a path computation) starts on a source TE TTP, goes through a chain of TE links and stops on a destination TE TTP.]

```
_____________________________________________________________________________

        |  |  |  +--rw source?                  inet:ip-address
        |  |  |  +--rw destination?             inet:ip-address
        |  |  |  +--rw src-tp-id?               binary
        |  |  |  +--rw dst-tp-id?               binary
_____________________________________________________________________________
```

- **TE tunnel hand-off point** – an access link or inter-domain link by which a multi-domain TE tunnel enters or exits a given network domain, in conjunction with a layer network resource (such as a wavelength channel or ODUk container) allocated on the access/inter-domain link for the TE tunnel.

- **TE tunnel segment** – a part of a multi-domain TE tunnel that spans a given network domain and is directly and fully controlled by the domain's controller, DC. TE tunnel segment is a fragment of a multi-domain TE tunnel between

  1. the source tunnel termination point and the TE tunnel hand-off point outbound from the TE tunnel's first domain (head TE tunnel segment);

  2. inbound and outbound TE tunnel hand-off points into/from a given domain (transit TE tunnel segment);

3. inbound TE tunnel hand-off point into the TE tunnel's last domain and the destination tunnel termination point (tail TE tunnel segment);

- **Transport service** - the same as TE tunnel segment

- **Hierarchy TE tunnel** - a server layer TE tunnel that supports a dynamically created TE link in the client layer network topology (e.g. see 1.2)

```
/* Hierarchy TE tunnel parameters */
      |  |  |  +--rw hierarchical-link-id
      |  |  |  |  +--rw local-te-node-id?      te-types:te-node-id
      |  |  |  |  +--rw local-te-link-tp-id?   te-types:te-tp-id
      |  |  |  |  +--rw remote-te-node-id?     te-types:te-node-id
      |  |  |  |  +--rw te-topology-id?        te-types:te-
topology-id
```

- **Hierarchy transport service** – the first or the last segment of a multi-domain hierarchy TE tunnel

- **Potential TE tunnel/segment** – a TE tunnel/segment configured in COMPUTE_ONLY mode. For such a TE tunnel/segment TE paths to be taken by supporting connection(s) is/are computed and monitored, but the connection(s) are not provisioned

```
      |  |  |     |  +--rw path-computation-method?   identityref
      |  |  |     |  +--rw path-computation-server?   inet:ip-
address
      |  |  |     |  +--rw compute-only?              empty
      |  |  |     |  +--rw use-cspf?                  Boolean
```

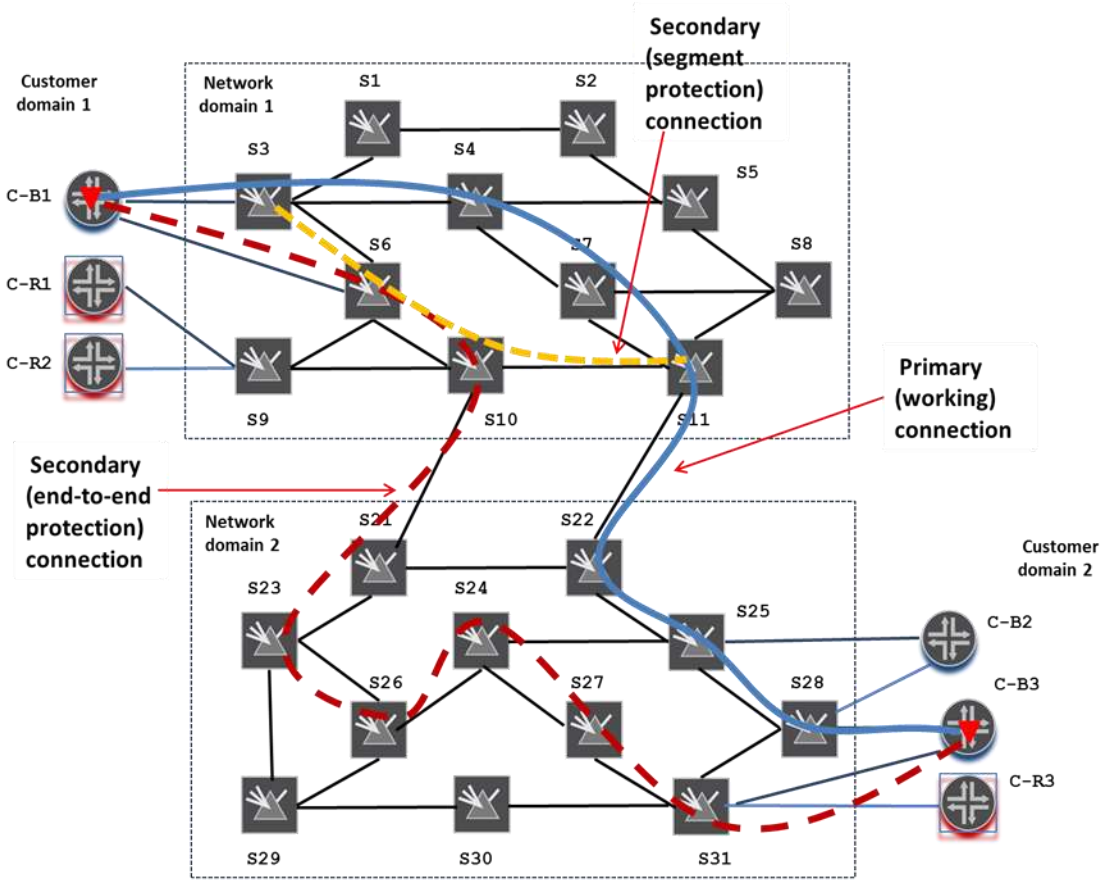

Figure 20a. TE Tunnel Connections/LSPs

- **Layer network connection/connection/LSP** - a layer network path supporting a TE tunnel by realizing its implied forwarding function. Said path is provisioned in a given layer network's data plane over a chain of links and cross-connected over switches terminating the links. It interconnects the supported TE tunnel's source and destination termination points (in the case of end-to-end connection) or TE tunnel's hand-off points (in the case of transport service connection) or the TE tunnel's two split-merge points (in the case of segment protection connection.

  Example: ODU2 connection supporting an ODU2 TE tunnel.

```
/* LSP (provisioned path) */
```

```
      |  |  |     |     +--ro lsp* [source destination tunnel-id
   lsp-id extended-tunnel-id type]
                    /* LSP parameters */
      |  |  |     |        +--ro source                leafref
      |  |  |     |        +--ro destination           leafref
      |  |  |     |        +--ro tunnel-id             leafref
      |  |  |     |        +--ro lsp-id                leafref
      |  |  |     |        +--ro extended-tunnel-id    leafref
      |  |  |     |        +--ro type                  leafref
      |  |  |     |        +--ro signaling-type?       identityref
   ..............................................................
      |  |  |              +--ro priority?              uint16
      |  |  |              +--ro path-setup-protocol?
   identityref
      |  |  |              +--ro active?                Boolean
```

- **Working connection** – the primary connection of the supported TE tunnel or transport service (see Figure 20a).

- **End-to-end protection connection** – a secondary end-to-end connection of the supported TE tunnel (e.g. end-to-end 1+1 protection connection, see Figure 20a).

- **Segment protection connection** – a secondary connection of the supported transport service protecting the service over a given network domain (e.g. 1+1 segment protection connection, see Figure 20a)

- **Unprotected TE tunnel/transport service** – TE tunnel/transport service supported by a single (working/primary) connection/LSP

- **Protected TE tunnel/transport service** – TE tunnel/transport service supported by one working connection/LSP and at least one protection/secondary connection/LSP

- **Connection configured path** – a TE path (see 1.2) over a TE topology describing a layer network/domain that specifies (loosely or strictly) the client’s requirements with respect to an ordered list of network nodes, links and resources on the links a given connection should go through

```
   |  |        +--rw explicit-route-object* [index]
         |  |           +--rw index                    leafref
         |  |           +--rw explicit-route-usage?   identityref
   (INCLUDE/EXCLUDE)
         |  |           |  +--rw index?            uint32
         |  |           |  +--rw (type)?
         |  |           |     +--:(numbered)
         |  |           |     |  +--rw numbered-hop
         |  |           |     |     +--rw address?    te-types:te-tp-
   id
         |  |           |     |     +--rw hop-type?   te-hop-type
         |  |           |     +--:(as-number)
         |  |           |     |  +--rw as-number-hop
         |  |           |     |     +--rw as-number?   binary
         |  |           |     |     +--rw hop-type?    te-hop-type
         |  |           |     +--:(unnumbered)
         |  |           |     |  +--rw unnumbered-hop
         |  |           |     |     +--rw node-id?      te-types:te-
   node-id
         |  |           |     |     +--rw link-tp-id?   te-types:te-
   tp-id
         |  |           |     |     +--rw hop-type?     te-hop-type
         |  |           |     +--:(label)
         |  |           |     |  +--rw label-hop
         |  |           |     |     +--rw value?   rt-
   types:generalized-label
         |  |           |     +--:(sid)
         |  |           |        +--rw sid-hop
         |  |           |           +--rw sid?   rt-
   types:generalized-label
```

- **Connection exclusion path** – a TE path over a TE topology describing a layer network/domain that specifies the client's requirements with respect to an unordered list of network nodes, links and resources on the links to be avoided by a given connection

---

```
   |        |  +--rw route-object-exclude-always* [index]
         |        |  |  +--rw index     leafref
         |        |  |  |  +--rw index?            uint32
         |        |  |  |  +--rw (type)?
```

```
          |        |  |  |     +--:(numbered)
          |        |  |  |     |  +--rw numbered-hop
          |        |  |  |     |     +--rw address?    te-types:te-tp-
     id
          |        |  |  |     +--:(as-number)
          |        |  |  |     |  +--rw as-number-hop
          |        |  |  |     |     +--rw as-number?   binary
          |        |  |  |     +--:(unnumbered)
          |        |  |  |     |  +--rw unnumbered-hop
          |        |  |  |     |     +--rw node-id?      te-types:te-
     node-id
          |        |  |  |     |     +--rw link-tp-id?   te-types:te-
     tp-id
          |        |  |  |     +--:(label)
          |        |  |  |     |  +--rw label-hop
          |        |  |  |     |     +--rw value?   rt-
     types:generalized-label
          |        |  |  |     +--:(sid)
          |        |  |  |        +--rw sid-hop
          |        |  |  |           +--rw sid?   rt-
     types:generalized-label
```

- **Connection computed path** – a TE path over a TE topology describing a layer network/domain as computed (subject to all configured constraints and optimization criteria) for a given connection to take. Computed connection path could be thought as the TE path intended to be taken by the connection

---

```
     /* Computed path */
                      /* Computed path properties/metrics /
          |  |  |     |  +--ro computed-path-properties
          |  |  |     |  |  +--ro path-metric* [metric-type]
          |  |  |     |  |  |  +--ro metric-type           identityref
          |  |  |     |  |  |  +--ro accumulative-value?   uint64
                      /* Computed path affinities */
          |  |  |     |  |  +--ro path-affinities
          |  |  |     |  |  |  +--ro constraints* [usage]
          |  |  |     |  |  |     |  +--ro usage?
     identityref
          |  |  |     |  |  |     |  +--ro (style)?
          |  |  |     |  |  |     |     +--:(value)
```

```
        |  |  |     |  |  |     |     |  +--ro value?            te-
   types:admin-groups
        |  |  |     |  |  |     |     +--:(named)
        |  |  |     |  |  |     |        +--ro affinity-names*
   [name]
        |  |  |     |  |  |     |           +--ro name    string
                  /* Computed path SRLGs */
        |  |  |     |  |  +--ro path-srlgs
        |  |  |     |  |  |  +--ro (style)?
        |  |  |     |  |  |     +--:(values)
        |  |  |     |  |  |     |  +--ro usage?         identityref
        |  |  |     |  |  |     |  +--ro values*        te-
   types:srlg
        |  |  |     |  |  |     +--:(named)
        |  |  |     |  |  |        +--ro constraints* [usage]
        |  |  |     |  |  |           +--ro usage
   identityref
        |  |  |     |  |  |           +--ro constraint
        |  |  |     |  |  |              +--ro srlg-names* [name]
        |  |  |     |  |  |                 +--ro name    string
                  /* Computed path sub-objects */
        |  |  |     |  |  +--ro path-computed-route-objects
   ...................................................
```

- **Connection actual path** – an active connection's path as provisioned in the layer network's data plane in the form of a TE path over a TE topology describing the layer network/domain

1.8. Transport Service Mapping

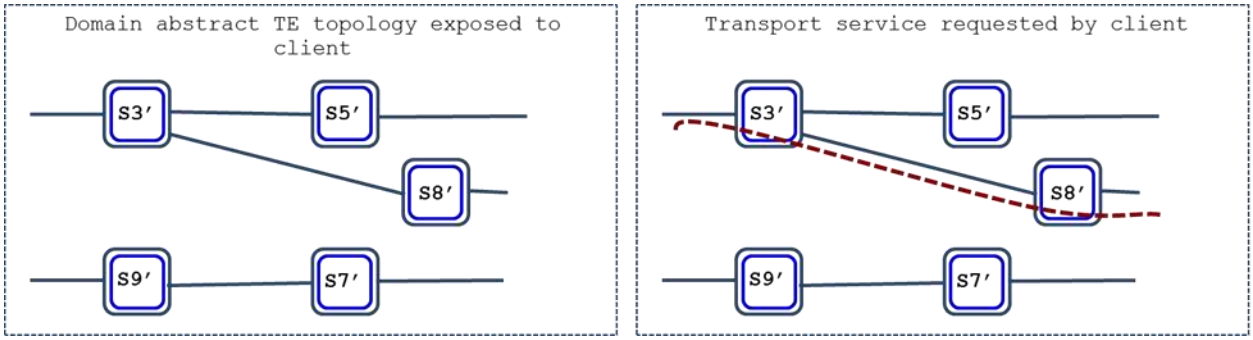

Figure 21. Transport Service Mapping

Let’s assume that a provider has exposed to a client its network domain in the form of an abstract TE topology, as shown on the left side of Figure 21. From then on, the provider should be prepared to receive from the client, a request to set up or manipulate a transport service with TE path(s) computed for the service connection(s) based on and expressed in terms of the provided abstract TE topology (as, for example, displayed in red broken line on the right side of Figure 21). When this happens, the provider is expected to set up the TE tunnels supporting all yet uncommitted abstract TE links (e. g, TE link S3’-S8’ in the Figure).

Furthermore, it is the responsibility of the provider to:

- Perform all the necessary abstract-to-native translations for the specified TE paths (i.e. the transport service connection configured paths);
- Provision working and protection connections supporting the transport service; as well as replace/modify/delete them in accordance with subsequent client’s configuration requests;
- Perform all the requested recovery operations upon detecting network failures affecting the transport service;
- Notify the client about all parameter changes, events and other telemetry information the client has expressed an interest in, with respect to the transport service in question.

1.9. Multi-Domain Transport Service Coordination

A client of multiple TE network domains may need to orchestrate/coordinate its transport service setup/manipulation across some or all the domains. One example of such a client is a Hierarchical T-SDN Controller, HC, managing a connected multi-domain transport network where each of the domains is controlled by a separate Domain T-SDN Controller, DC. Said DCs are expected to expose TE Topology and TE Tunnel North Bound Interfaces, NBIs,, supported respectively by IETF TE Topology and TE Tunnel models (and their network layer specific augmentations). HC is assumed to establish client-provider relationship with each of the DCs and make use of said NBIs to extract from the domains various information (such as TE topologies and telemetry), as well as to convey instructions to coordinate across multiple domains its transport services set up and manipulation.

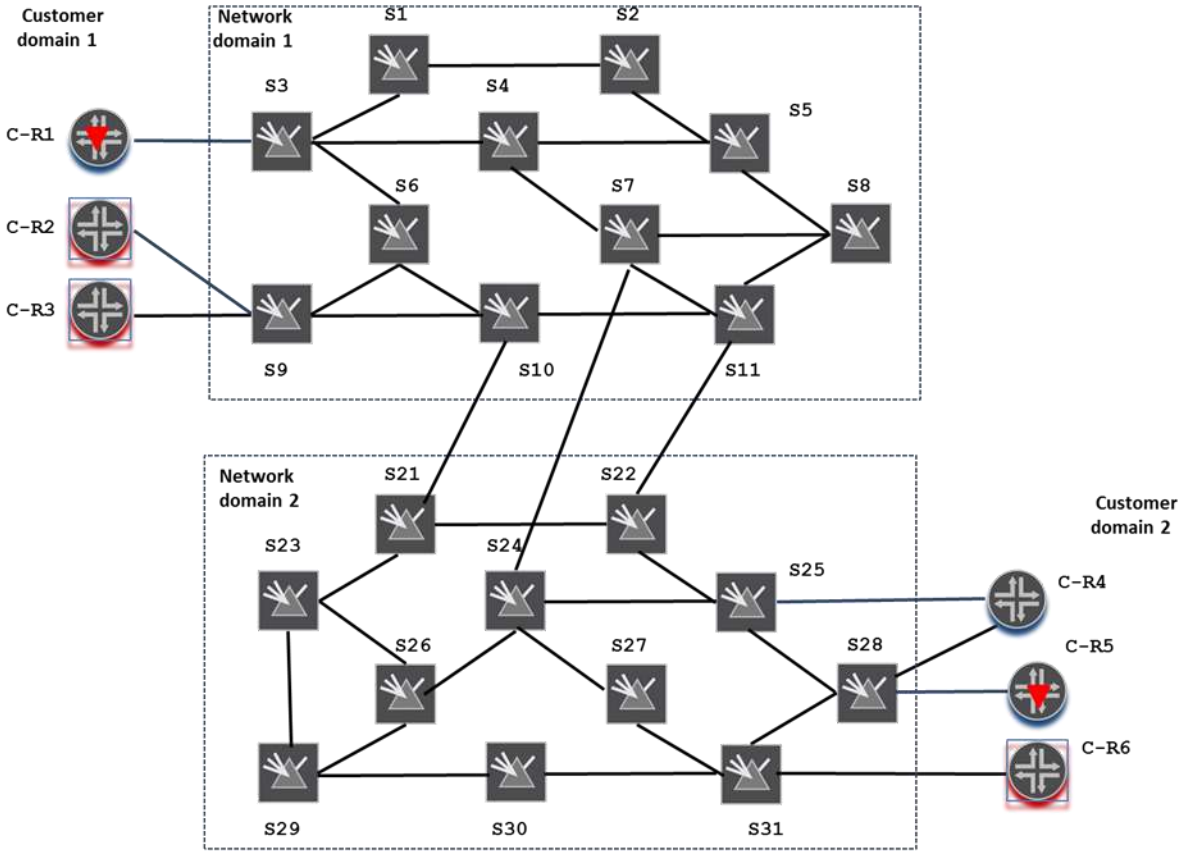

Figure 22. Two-Domain Transport Network

Let's consider, for example, a two-domain transport network as represented in Figure 22. Suppose that HC is requested to set up an unprotected transport service to provide connectivity between customer network elements C-R1 and C-R6. It is assumed that by the time the request has arrived, the two DCs have already provided abstract TE topologies describing their respective domains, and that HC has merged the provided TE topologies into one that homogeneously describes the entire transport network (as shown in Figure 23).

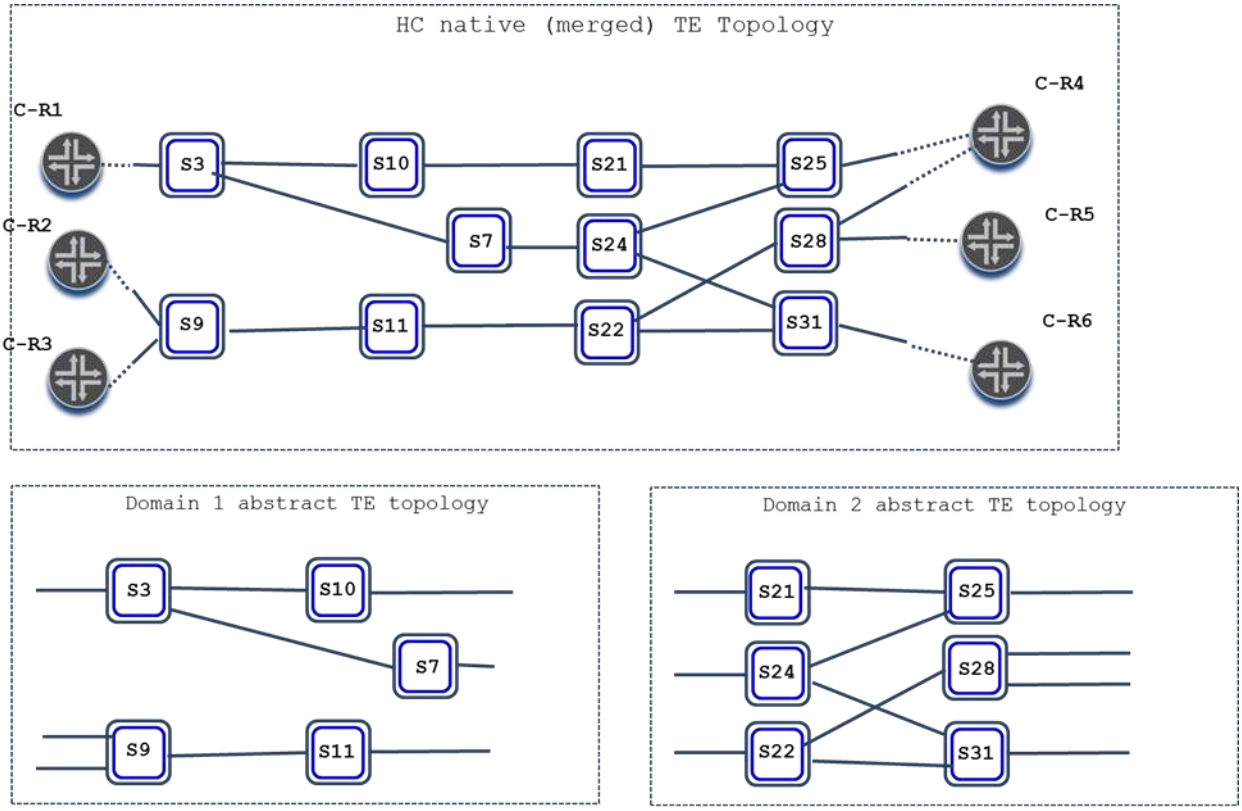

Figure 23. Two-Domain Transport Network (Abstracted View)

Consider that HC, using the merged TE topology, selected a TE path to be taken by the requested transport service connection as shown on the upper part of Figure 24.

The multi-domain transport service set up coordination includes:

- Splitting selected for the transport service TE path(s) into segments – one set of segments per each domain involved in the service setup;

- Issuing a configuration request to each of the involved DCs to set up the transport service across the respective domain. Note that the connection configured paths are required to be expressed in terms of respective abstract TE topologies as exposed to HC by DCs (see lower part of Figure 24).

- Waiting for the set up complete confirmation from each of the involved DCs. In case one of the DCs reports a failure, HC is responsible to carry out the cleanup/rollback procedures by requesting all involved DCs to tear down the successfully created segments

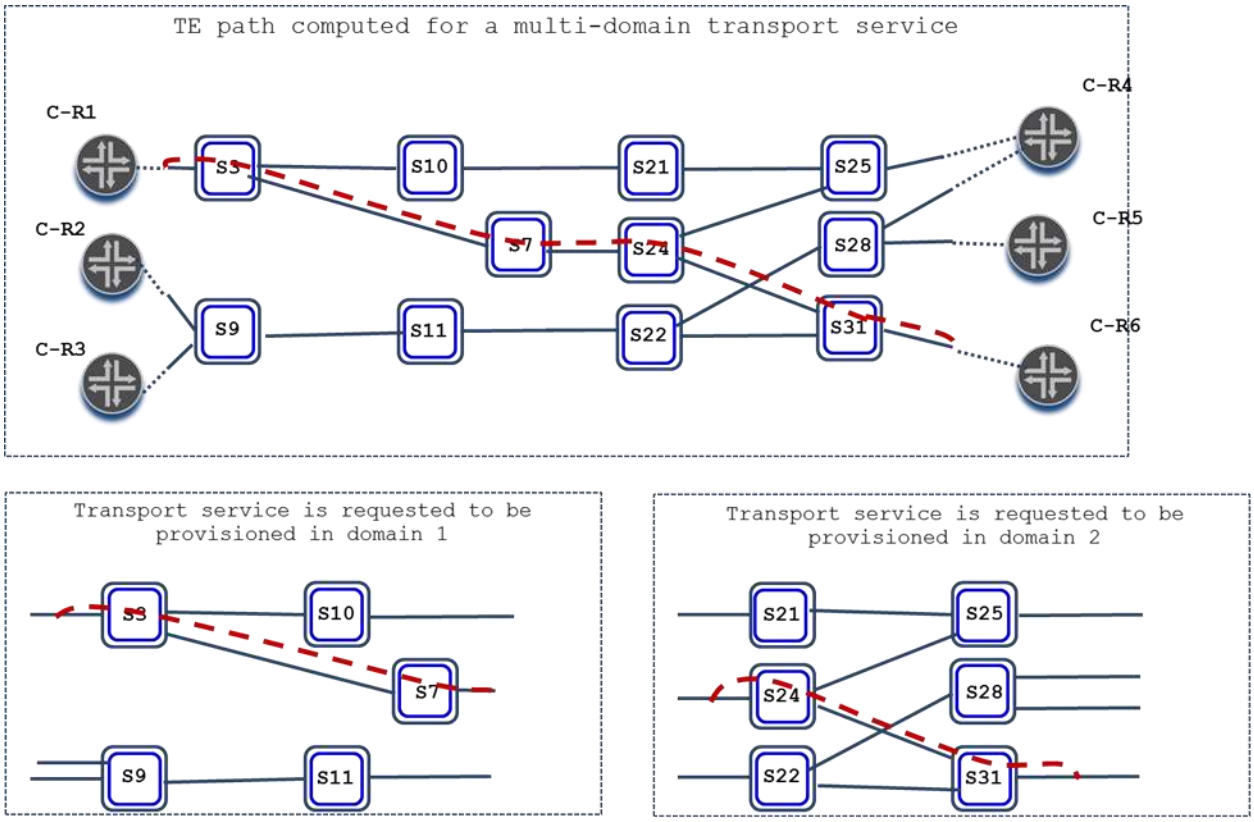

Figure 24. Transport Service Placement Based on Abstract TE Topology

While processing the received from HC configuration request to set up the transport service, each DC is expected to carry out the transport service mapping procedures (as described in 1.8) resulting in the set up of all the necessary underlay TE tunnels, as well as one or more connections supporting the transport service. As a result, the requested transport service will be provisioned as shown in Figure 25.

The multi-domain transport service tear down coordination entails issuing to each of the involved DCs a configuration request to delete the transport service in the controlled by the DC domain. DCs are

expected in this case to release all network resources allocated for the transport service.

The multi-domain transport service modify coordination implies issuing to each of the involved DCs a configuration request to replace the transport service connections according to the newly provided paths and/or modify the connection parameters according to the newly provided configuration.

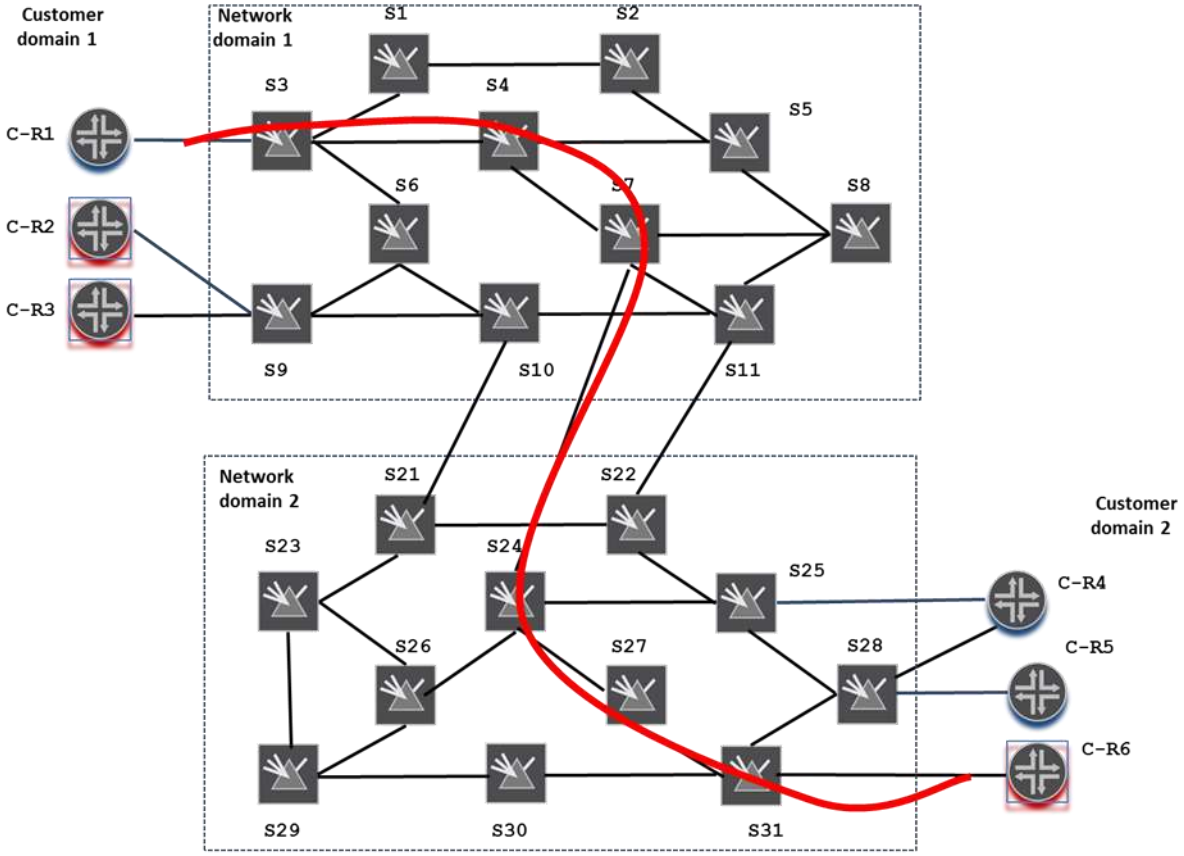

Figure 25. Multi-domain transport service is provisioned

2. Use Cases

2.1. Use Case 1. Transport service control on a single layer multi-domain transport network

Configuration (Figure 26):

- Three-domain multi-vendor ODUk/Och transport network;
- The domains are interconnected via ODUk inter-domain links;
- Each of the domains is comprised of ODUk/Och network elements (switches) from a separate vendor and is controlled by a single (vendor specific) T-SDN Domain Controller (DC);
- All DCs expose IETF TE Topology and TE Tunnel model based NBIs;
- The transport network as a whole is controlled by a single hierarchical T-SDN controller (HC);
- HC makes use of the NBIs to set up client-provider relationship with each of the DCs and controls via the DCs their respective network domains;
- Three customer IP/MPLS sites are connected to the transport network via ODUk access links;
- The customer IP/MPLS routers and the router transport ports connecting the routers to the transport network are managed autonomously and independently from the transport network.

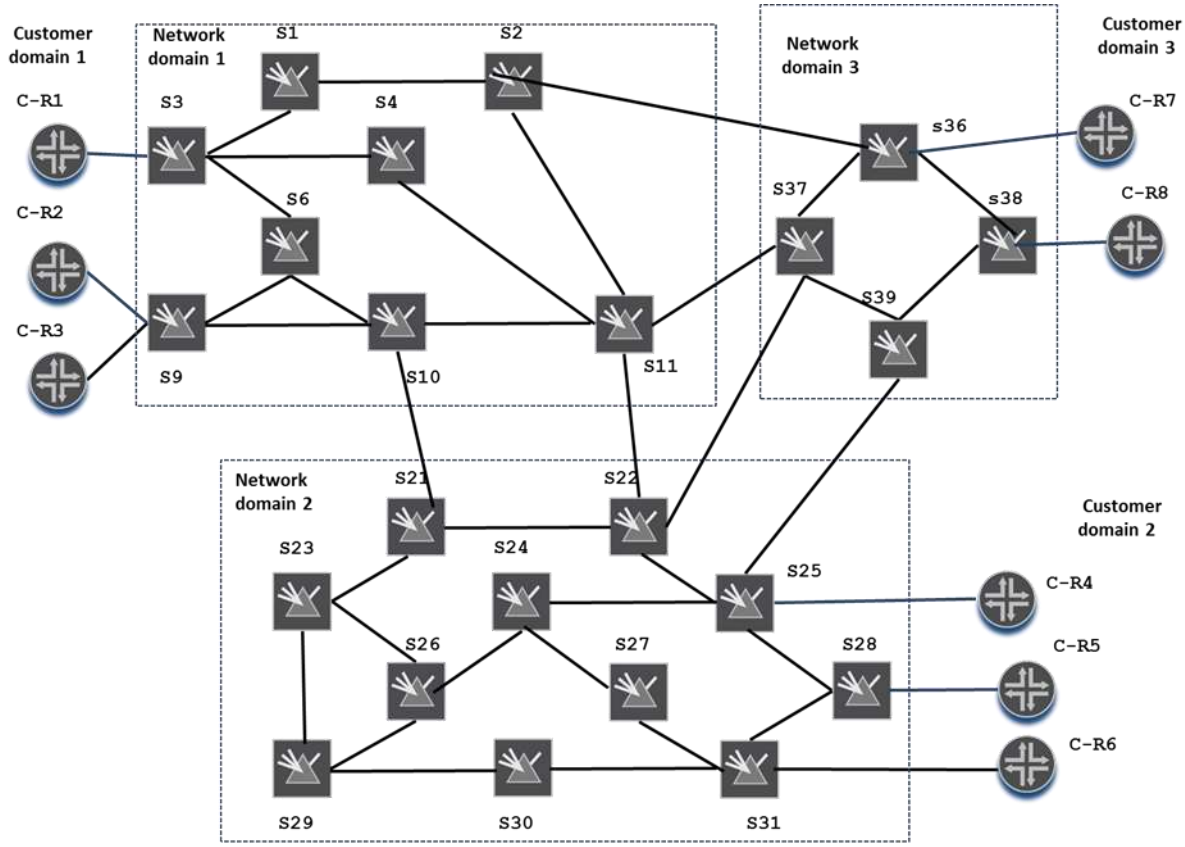

Figure 26 Three-domain ODUk/Och transport network with ODUk access and inter-domain links

Objective: Set up/manipulate/delete a shortest delay unprotected or protected transport service to provide connectivity between customer network elements C-R2 and C-R5

1) TE Topology discovery

All DCs provide to HC respective domain ODUk layer abstract TE topologies. Let's assume that each such topology is a single-node TE topology (as described in 1.3.1, abstract TE topology of this type represents the entire domain as a single asymmetrical/blocking TE node). Let's further assume that the abstract TE nodes representing the domains are attributed with detailed connectivity matrices optimized according to the shortest delay criterion. [Note: single-node abstract TE topologies are assumed for simplicity sake. Alternatively, any DC could have provided an abstract TE topology of any type described in 1.3].

HC merges the provided TE topologies into its own native TE topology (the TE topology merging procedures are discussed in 1.4). The merged TE topology, as well as the TE topologies provided by DCs, are depicted in Figure 27. The merged TE topology homogeneously describes the entire transport network and hence is suitable for path computations across the network. Note that the dotted lines in the Figure connecting the topology access TE links with customer devices illustrate that HC in this use case has neither control nor information on the customer devices/ports and, therefore, can only provide a connectivity between the requested transport service ingress and egress access links (on assumption that the customer transport ports are provisioned independently)

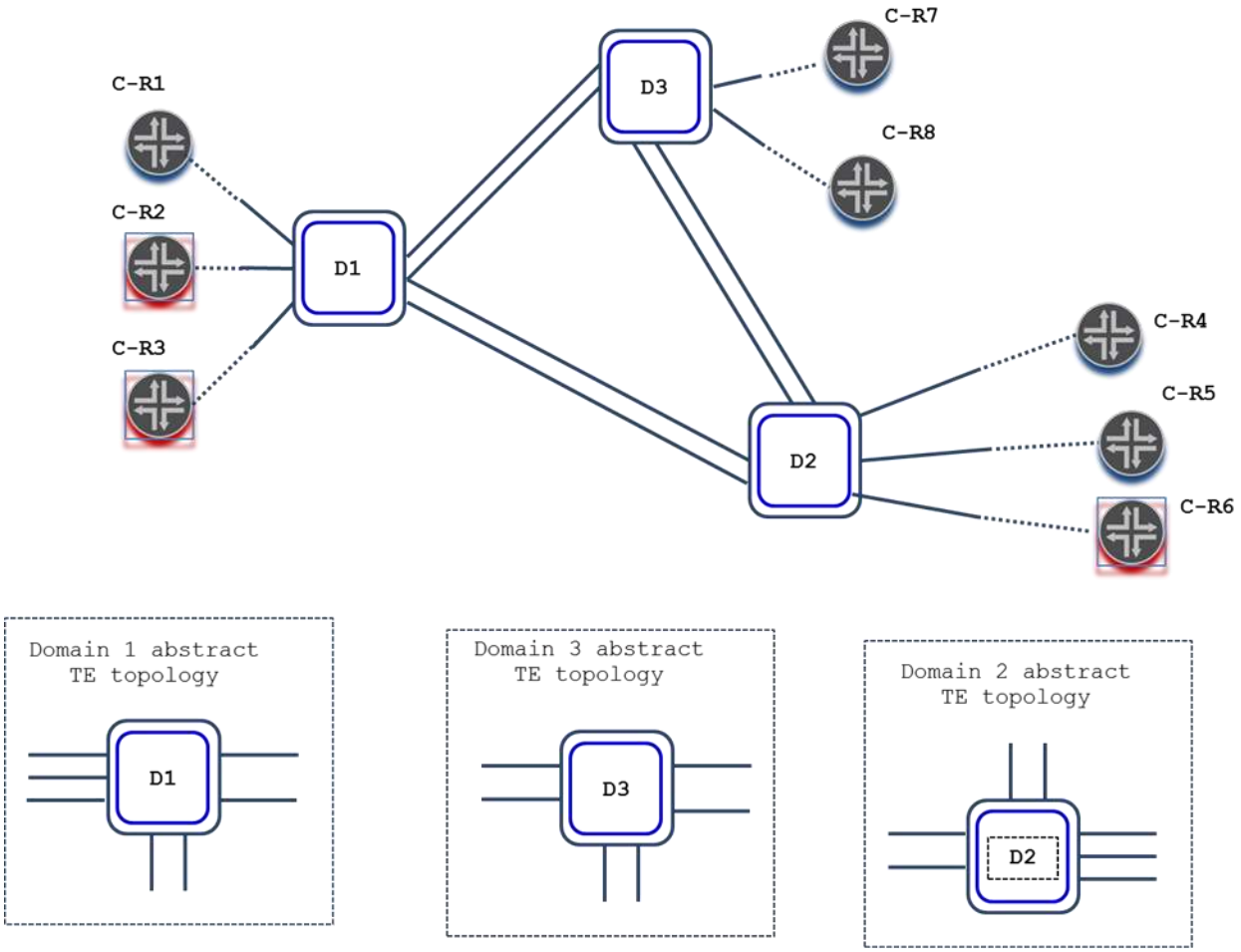

Figure 27. Three-domain single layer transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 27, upper part) HC selects one or more optimal and sufficiently disjoint from each other TE path(s) for the requested transport service connection(s). Resulting TE paths for the requested end-to-end protected transport service, for example, could be as marked on the upper part of Figure 28.

It is important to keep in mind that HC's path computer is capable of performing the necessary path selection only as long as the merged TE topology provides the necessary TE visibility for the path selection, both intra-domain (e.g. by virtue of provided by the abstract TE nodes detailed connectivity matrices) and inter-domain (because of provided inter-domain TE link attributes). In case one or more DCs is/are not capable of or willing to provide the detailed connectivity matrices (that is, DCs expose the respective domains as black boxes - unconstrained TE nodes terminating the inter-domain TE links), HC will not be able to select the end-to-end TE path(s) for the requested transport service on its own. In such a case HC may opt for making use of the Path Computation NBI, exposed by the DCs to explore/evaluate intra-domain TE path availability in real time. IETF TE Tunnel model supports the Path Computation NBI by allowing for the configuration of transport services in COMPUTE_ONLY mode. In this mode the provider is expected to compute TE paths for a requested transport service connections and return the paths in the request's response without triggering the connection provisioning in the network.

Consider, for example, the case when none of the DCs has provided the detailed connectivity matrix attribute for the abstract TE nodes representing the respective domain. In such a case HC may:

1. Request the ingress domain DC (i.e. DC1) to compute intra-domain TE paths connecting the ingress access TE link (i.e. the link facing C-R2) with each of the inter-domain TE links (i.e. links connecting Domain 1 to Domain 2 and Domain 3 respectively);

2. Grow the TE paths returned by DC1 in (1) over the respective outbound inter-domain TE links;

3. Request the neighboring DC(s) (e.g. DC3) to compute all intra-domain TE paths connecting across the domain all inbound into the domain inter-domain TE links reached by the path growing process in (2) with all other (outbound) domain's inter-domain TE links;

4. Augment the TE paths produced in step (2) with the TE paths determined in step (3);

5. Repeat steps (2), (3) and (4) until the resulting TE paths reach the egress domain (i.e. Domain 2);

6. Request the egress domain DC (i.e. DC2) to grow each of the TE paths across the domain to connect them to the egress access TE link (i.e. the link facing C-R5);

7. Select one (or more) most optimal and sufficiently disjoint from each other TE path(s) from the list produced in step (6).

[Note: The transport service path selection method based on Path Computation NBIs exposed by DCs does not scale well and the more domains comprise the network and the more inter-domain links interconnect them, the worse the method works. Realistically, this approach will not work sufficiently well for the networks with more than 3 domains]

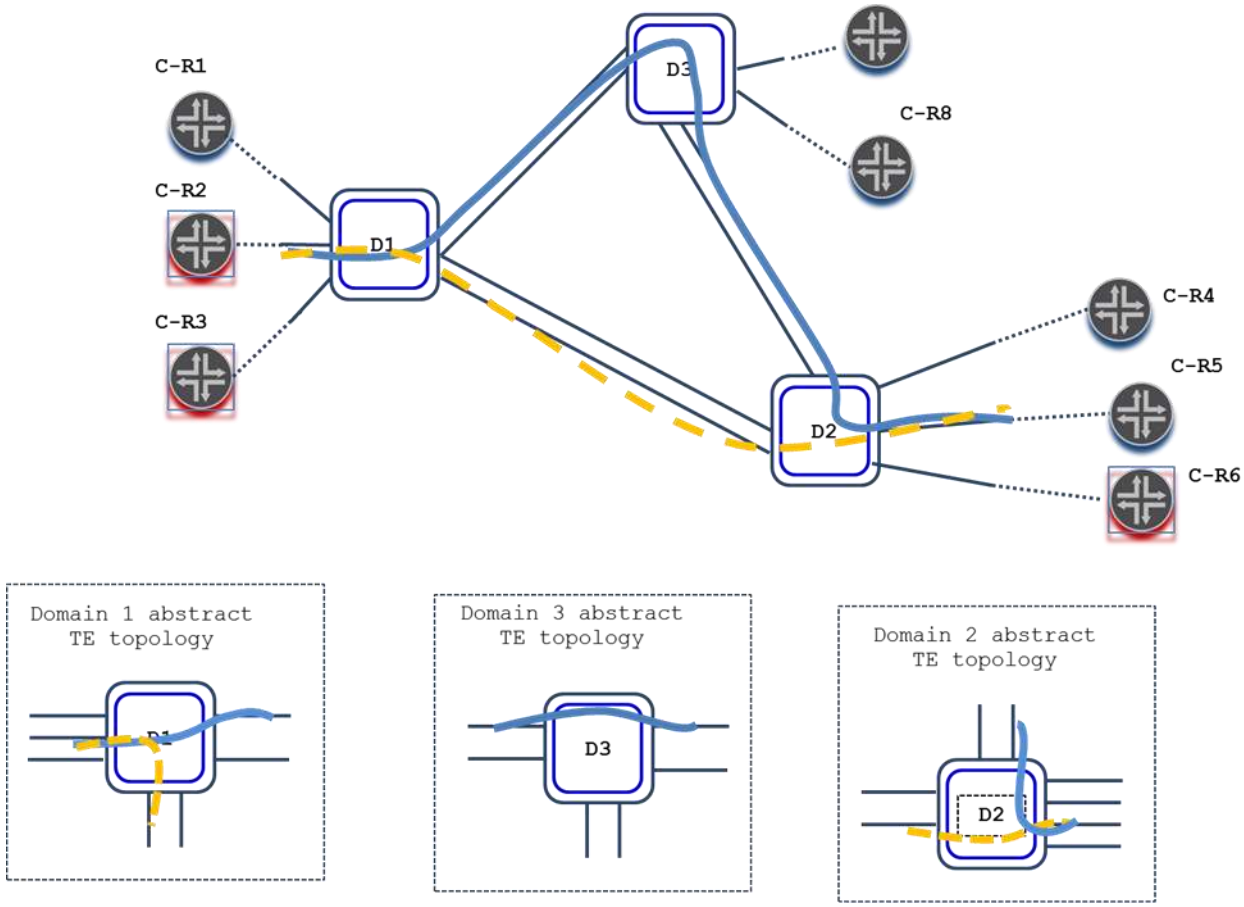

Figure 28. TE paths computed for the protected transport service

3) Transport service setup coordination

HC carries out the multi-domain transport service setup coordination as described in 1.9. In particular, HC splits the computed TE path(s) into 3 sets of TE path segments – one set per domain (as shown on the lower part of Figure 28), and issues a TE tunnel configuration request to each of the DCs to set up the requested transport service across the domain under the DC's control. The primary (and secondary) connection explicit path(s) is/are specified in the requests in terms of respective domain abstract TE topologies.

While processing the configuration request, each DC performs the transport service mapping (as described in 1.8). In particular, the DC translates the specified explicit path(s) from abstract into native TE topology terms, sets up supporting underlay TE tunnels

(e.g. Och TE tunnels), and, then, allocates required ODUk containers on the selected links and provisions the ODUk cross-connects on the switches terminating the links.

If the setup is successfully completed in all three domains, the transport service connection(s) will be provisioned as depicted in Figure 29. If one of the DCs fails to set up its part, all successfully provisioned segments will be asked by HC to be released.

4) Transport service teardown coordination

HC issues to each of DCs a configuration request to release the transport service over the controlled domain, as well as the server layer TE tunnels supporting dynamically created links.

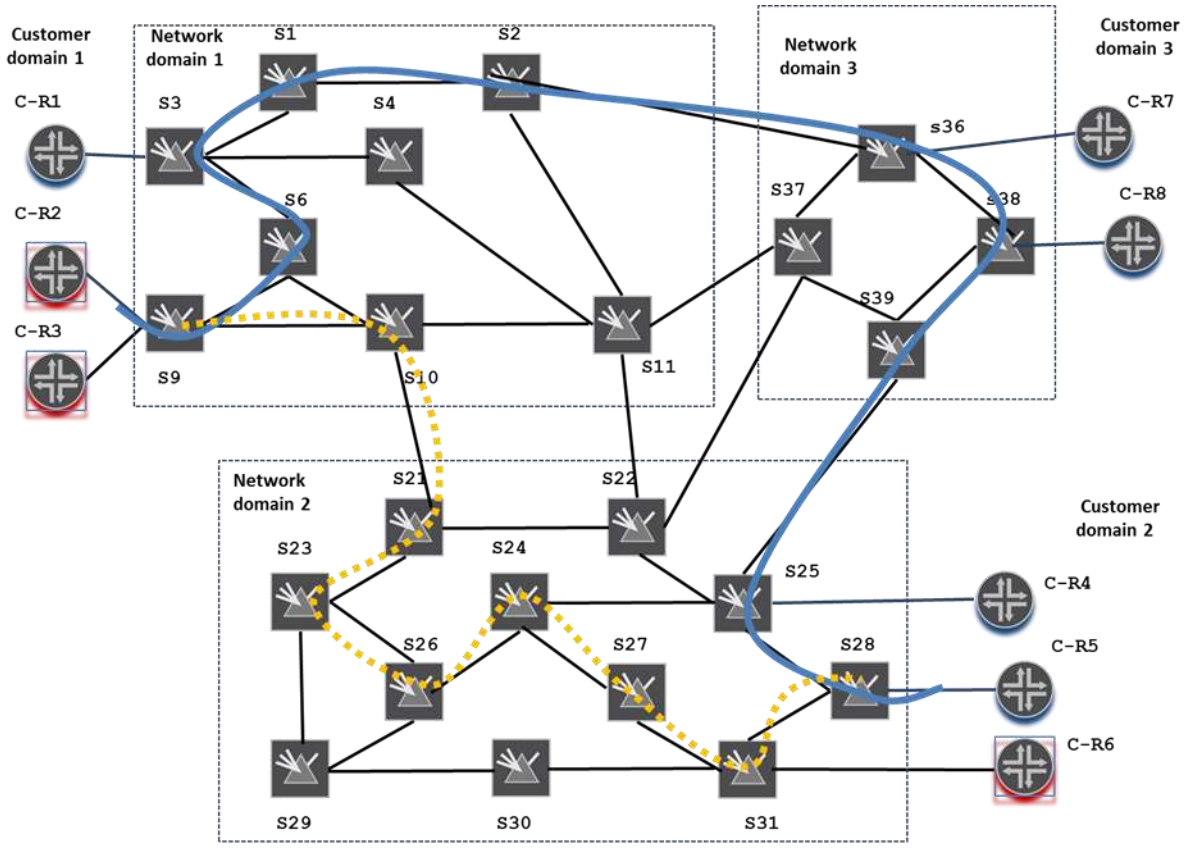

Figure 29. Transport service is provisioned

2.2. Use Case 2. End-to-end TE tunnel control on a single layer multi-domain transport network

Configuration (Figure 26): the same as in use case 1, except that HC in this use case controls customer devices/ports by extracting information from and pushing configuration to the customer site SDN controller(s) managing the customer devices directly.

Objective: Set up//delete an unprotected shortest delay TE tunnel interconnecting end-to-end C-R2 and C-R5

1) TE Topology discovery

As in use case 1 all DCs provide to HC domain ODUk layer abstract TE topologies. Additionally in this use the three customer site controllers expose the TE Topology and Tunnel model based NBIs to HC. Using the TE Topology NBI each customer controller provides to HC the respective customer site domain abstract TE topology. Customer site abstract TE topologies contain abstract TE nodes representing the devices which are directly connected to the transport network. Said abstract TE nodes host TE tunnel termination points, TTPs, representing the ports over which the customer devices are connected to the transport network, and terminate access TE links the TTPs are accessible from (see Figure 30).

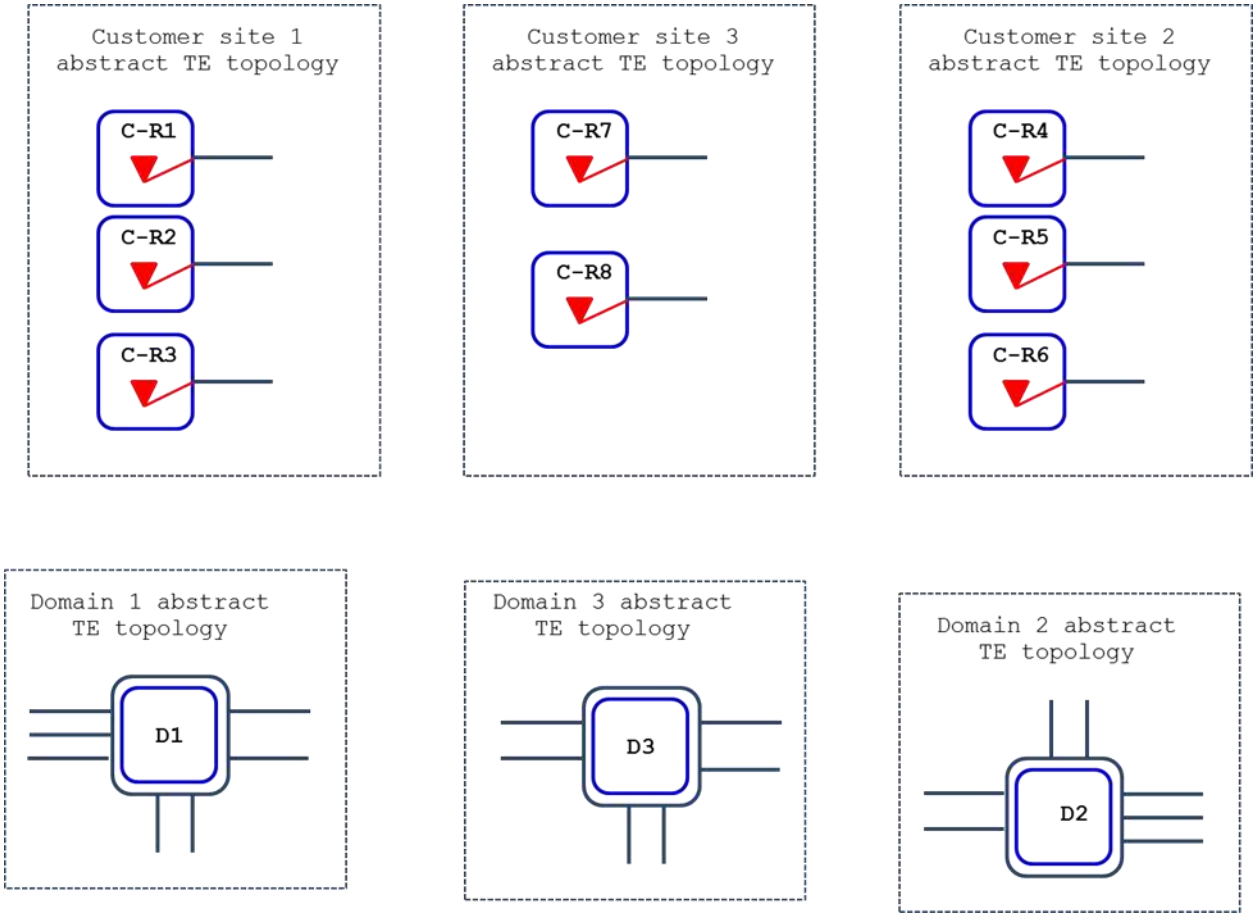

Figure 30. Abstract TE topologies provided by all network domains and customer sites

HC merges the provided topologies into its own native TE Topology (the TE topology merging procedures are discussed in 1.4). The merged TE topology is depicted in Figure 31. It homogeneously describes end-to-end not only the entire transport network, but also the customer sites connected to the network and hence is suitable for TE tunnel end to end path computations.

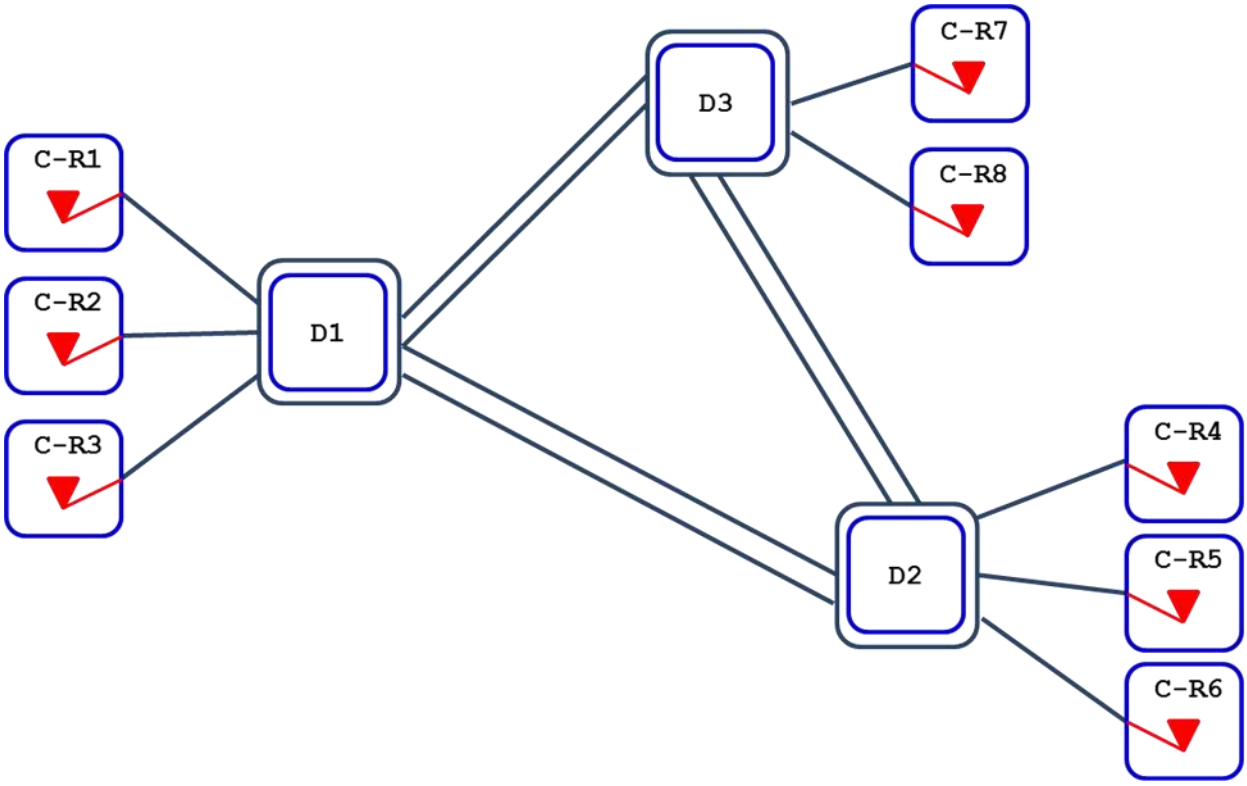

Figure 31. Abstract TE topology describing transport network and connected to it customer sites

2) TE tunnel path computation

Using the merged TE topology (Figure 31) HC selects an optimal TE path for the requested TE tunnel connecting end-to-end the specified TE tunnel termination points, TTPs. The resulting TE path, for example, could be as marked on the upper part of Figure 32.

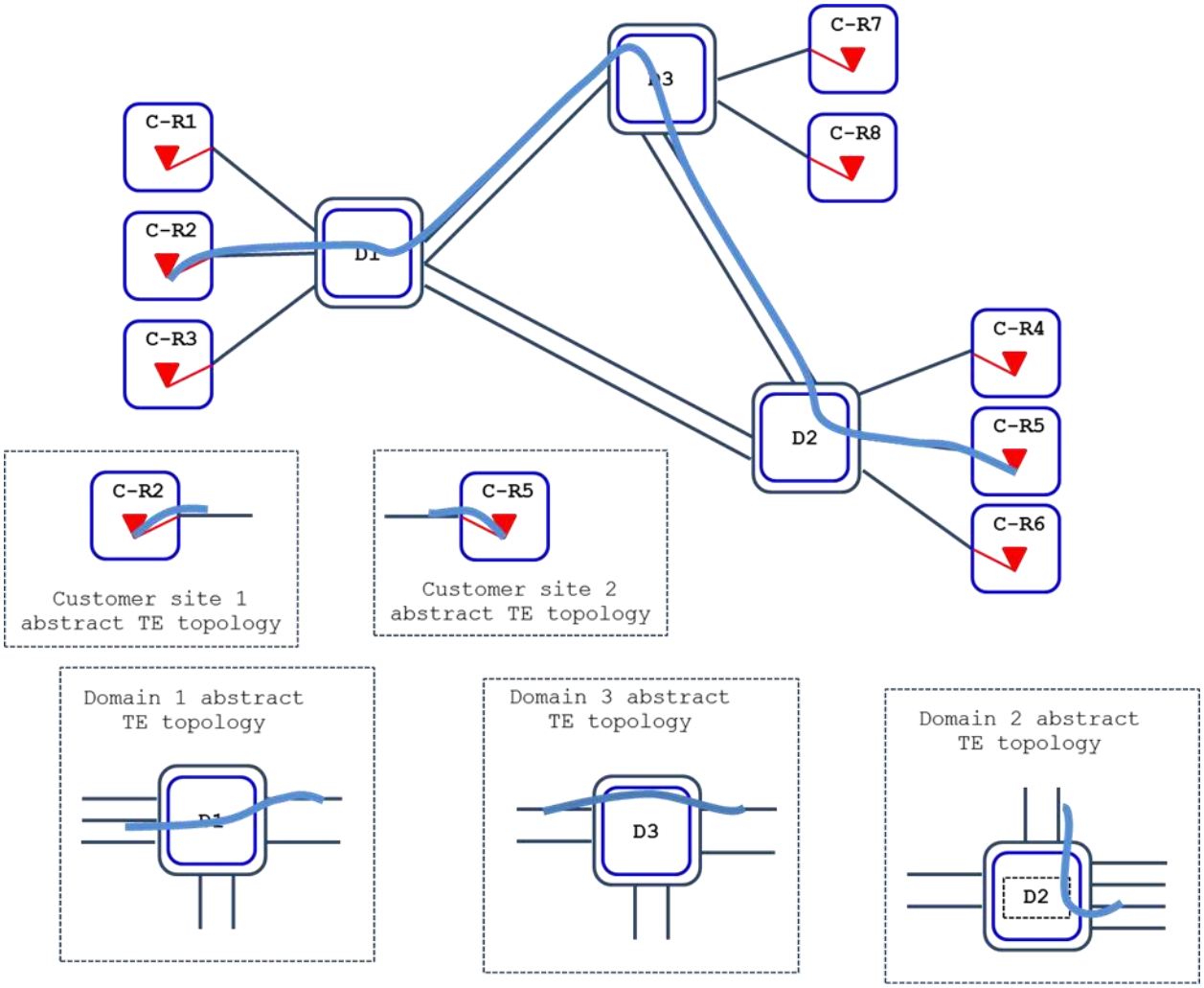

Figure 32. TE path computed for the TE tunnel

3) TE tunnel setup coordination

HC carries out the multi-domain TE tunnel setup coordination as described for use case 1, except that in this use case HC additionally initiates and controls the setup of the TE tunnel's head and tail segments on the respective customer sites. Note that the customer site controllers behave exactly as transport network domain DCs. In particular, they receive issued by HC configuration requests to set up the TE tunnel's head and tail segments respectively. While processing the requests the customer site controllers perform the necessary provisioning of the TE tunnel's source and destination termination points, as well as of the local sides of the selected

access links. If all segments are successfully provisioned on customer sites and network domains, the TE tunnel connection will be provisioned as marked in Figure 33.

4) TE tunnel teardown coordination

HC issues to each of DCs and customer site controllers a configuration request to release respective segments of the TE tunnel, as well as the server layer TE tunnels supporting dynamically created links.

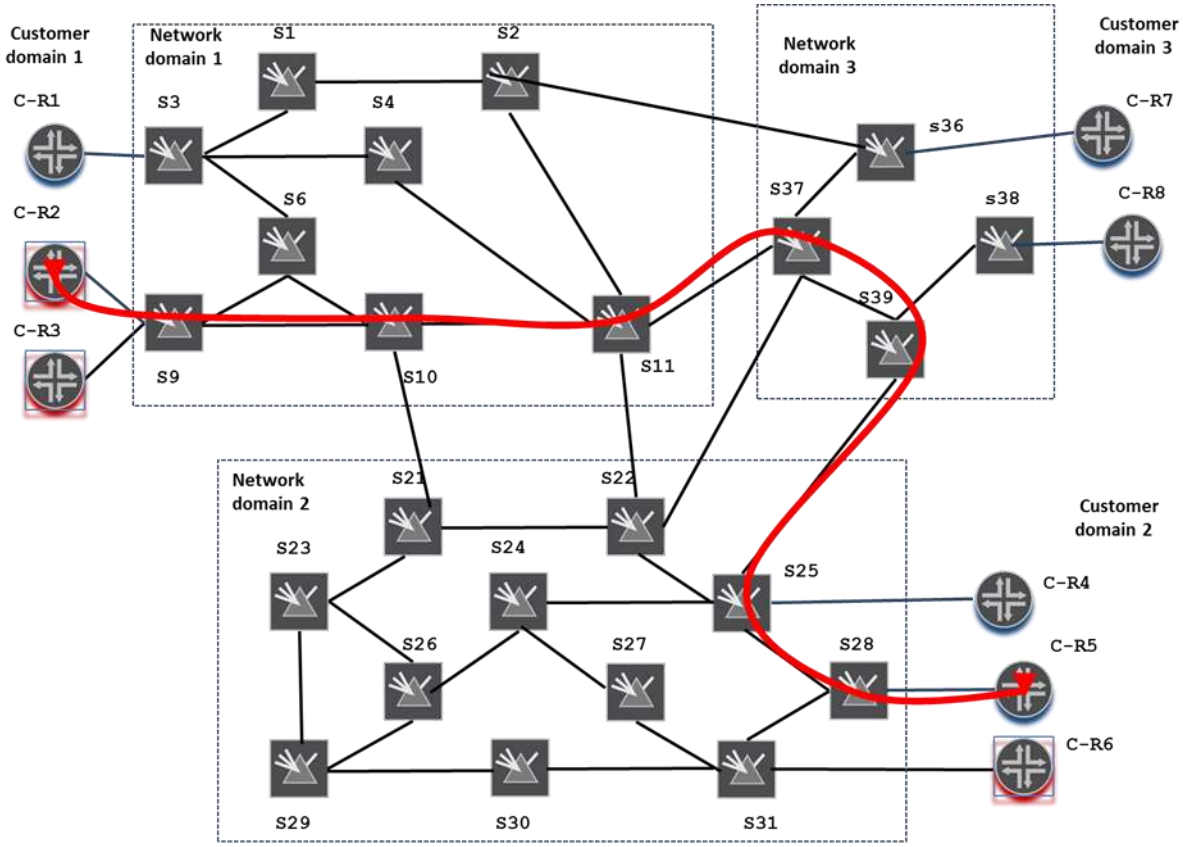

Figure 33. TE tunnel is provisioned

2.3. Use Case 3. Transport service control on a ODUk/Och multi-domain transport network with Ethernet access links

Configuration (Figure 34): the same as in use case 1, except that all access links in this use case are Ethernet layer links (depicted as

blue lines in the Figure), while all inter-domain links remain to be ODUk layer links.

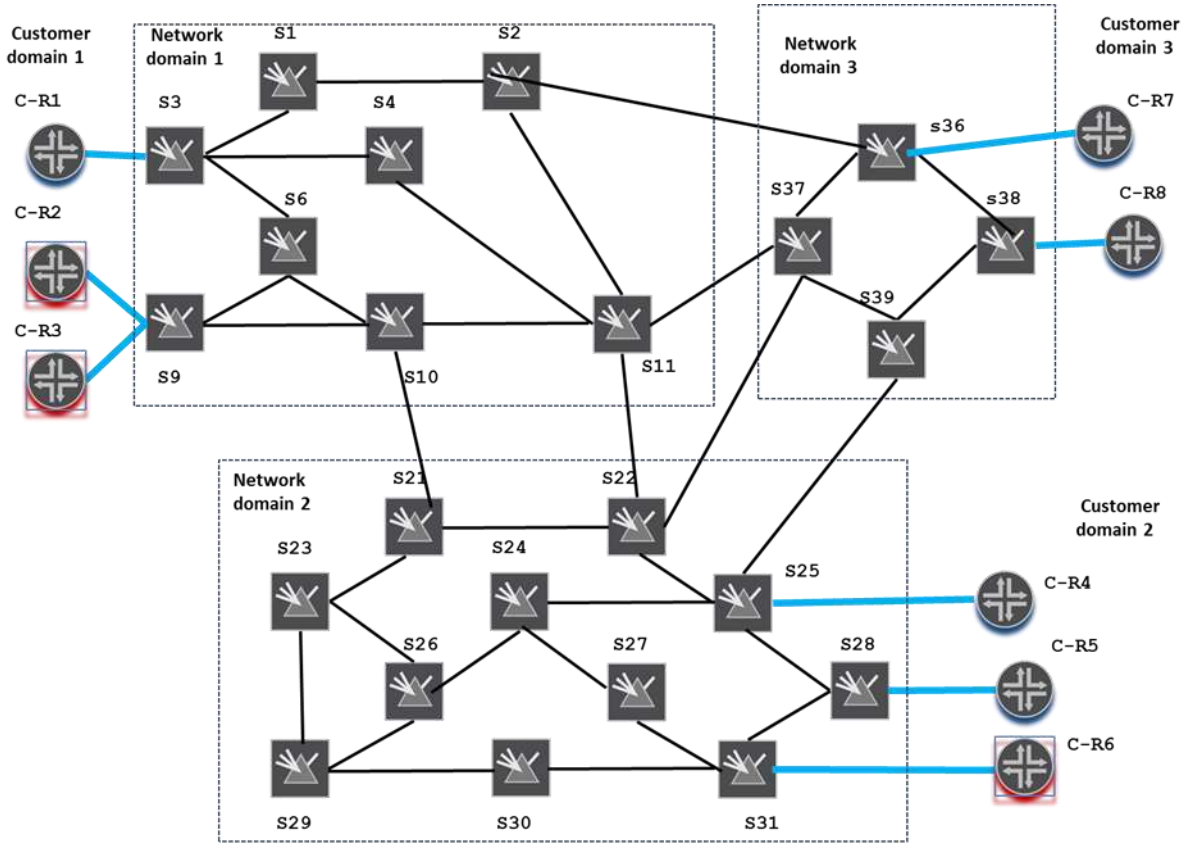

Figure 34. Three-domain ODUk/Och transport network with Ethernet layer access links

Objective: Set up//delete an unprotected shortest delay transport service supporting connectivity between C-R2 and C-R5

1) TE Topology discovery

In order to make possible for the necessary in this use case multi-layer path computation, each DC exposes to HC two (ODUk layer and Ethernet layer) abstract TE topologies, Additionally, the lower layer (ODUk) TE nodes announce hosted by them TE tunnel termination points, TTPs, capable of adopting the payload carried over the Ethernet layer access links, From the TE Topology model point of view this means that said TTPs are attributed with TE inter-layer locks

matching ones attributed to Ethernet TE links (i.e. TE links provided within Ethernet layer abstract TE topologies).

Ethernet and ODUk layer single node abstract TE topologies catered to HC by each of the DCs are presented in Figure 35.

HC merges the provided TE topologies into its own native TE Topology (the merging procedures are described in 1.4). Importantly in this case HC locks the provided TE topologies not only horizontally, but vertically as well, thus producing a two-layer TE topology homogenously describing both layers of the entire transport network, as well as the client-server layer adaptation relationships between the two layers. This makes the merged TE topology suitable for multi-layer/inter-layer multi-domain transport service path computations. The merged TE topology is presented in Figure 36.

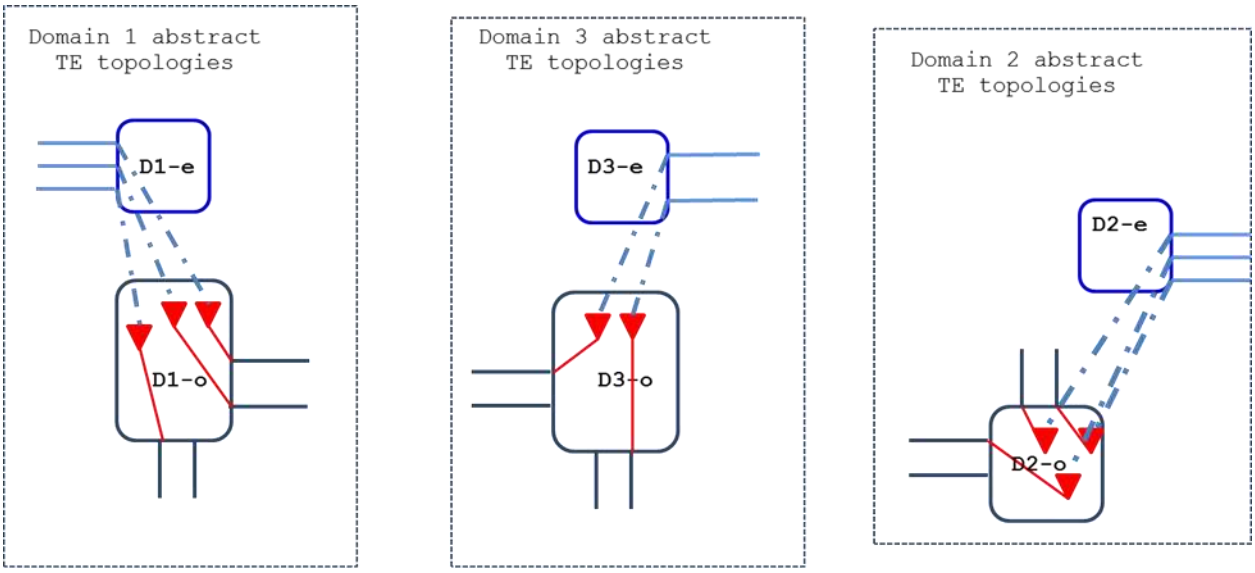

Figure 35. ODUk and Ethernet layer abstract TE topologies exposed by DCs

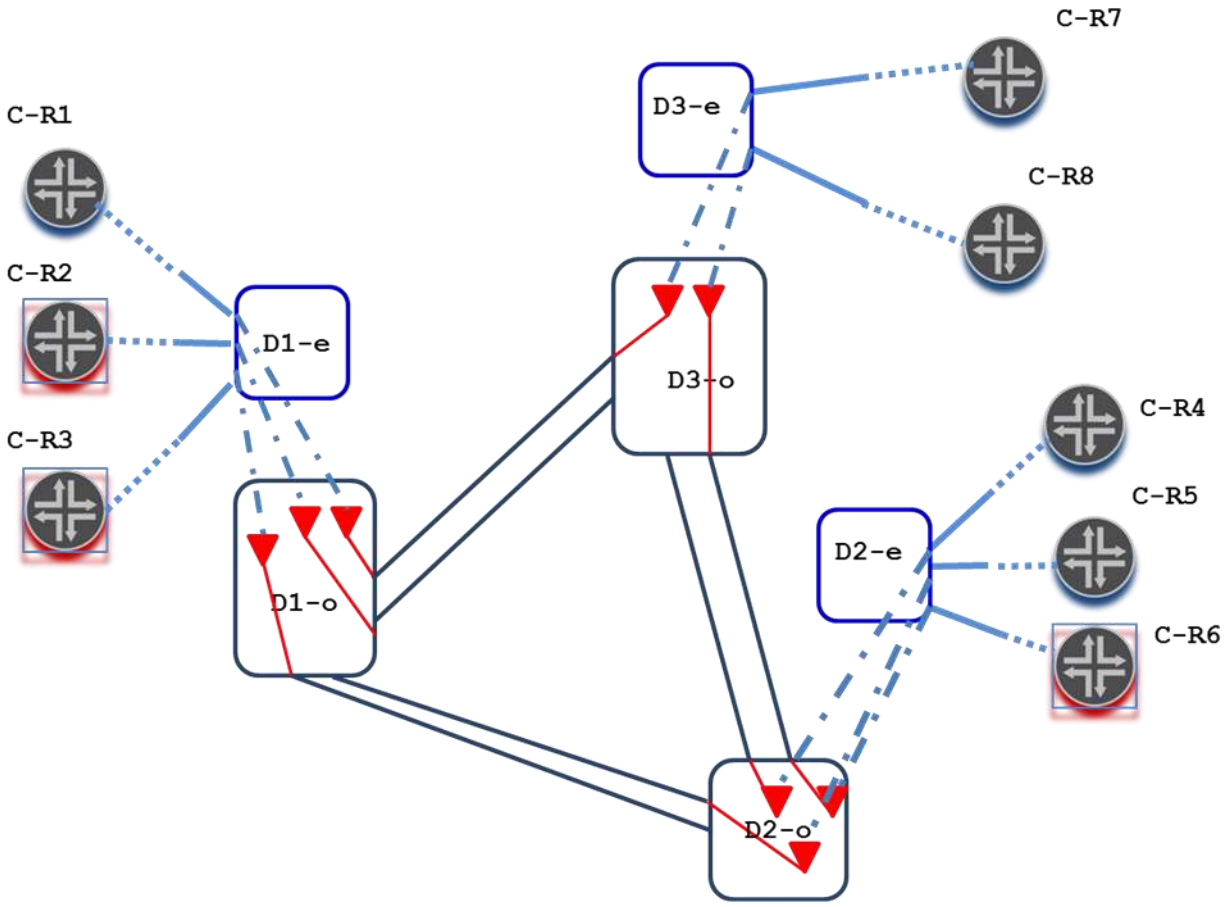

Figure 36. Two-layer three-domain transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 36) HC selects an optimal TE path for the requested transport service.

Note that if HC's path computer considered only Ethernet layer TE nodes and links, the path computation would .fail. This is because the Ethernet layer TE nodes (i.e. D1-e, D2-e and D3-e in the Figure) are disconnected from each other. However, the inter-layer associations (in the form of the TE inter-layer locks) make possible for the path computer to select TE path(s) in the lower (ODUk) layer that can be used to set up hierarchy TE tunnel(s) supporting additional dynamic TE link(s) in the upper (Ethernet) layer in order for the requested transport service path computation to succeed.

Let's sssume that the resulting TE path is as marked in Figure 37. The red line in the Figure marks the TE path selected for the ODUk layer hierarchy TE tunnel supporting the required Ethernet layer dynamic TE link.

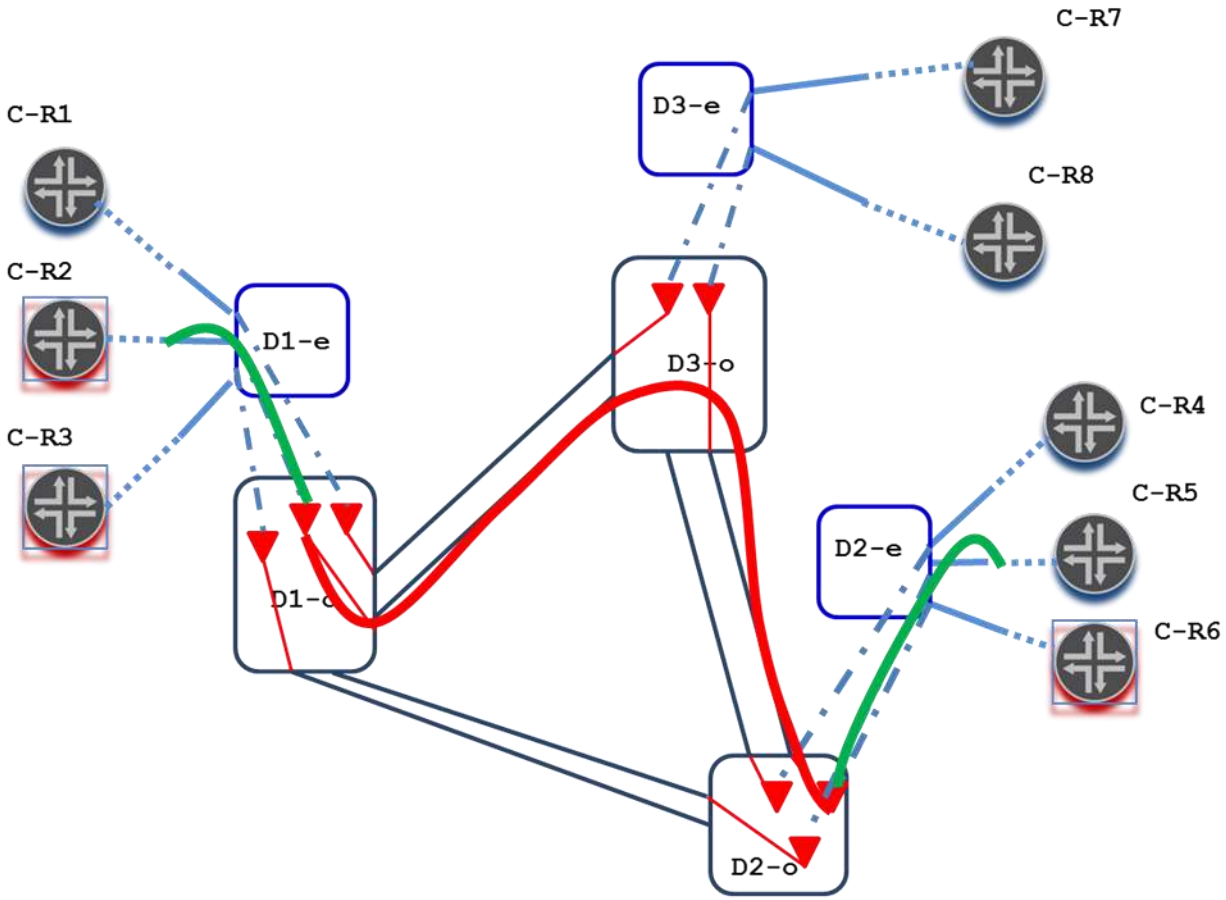

Figure 37. Multi-layer TE path computed for the transport service

3) Transport service setup coordination

HC sets up the requested Ethernet layer transport service in two stages. First, it coordinates the end-to-end setup of the ODUk layer hierarchy TE tunnel between the selected TTPs. If this operation succeeds, a new Ethernet layer dynamic TE link (blue line connecting TE nodes D1-e and D2-e in Figure 38) is automatically added to the merged abstract TE topology. Importantly, as a part of the hierarchy transport service setup both DC1 and DC 2 add a new open-ended Ethernet layer inter-domain dynamic TE link to their respective abstract TE topologies. Second, HC coordinates the setup of the requested (Ethernet layer) transport service. The required TE path for the second stage is marked as fat blue line in the Figure. Note that DC3 controlling domain 3 is only involved in the first stage, but is oblivious to the second stage.

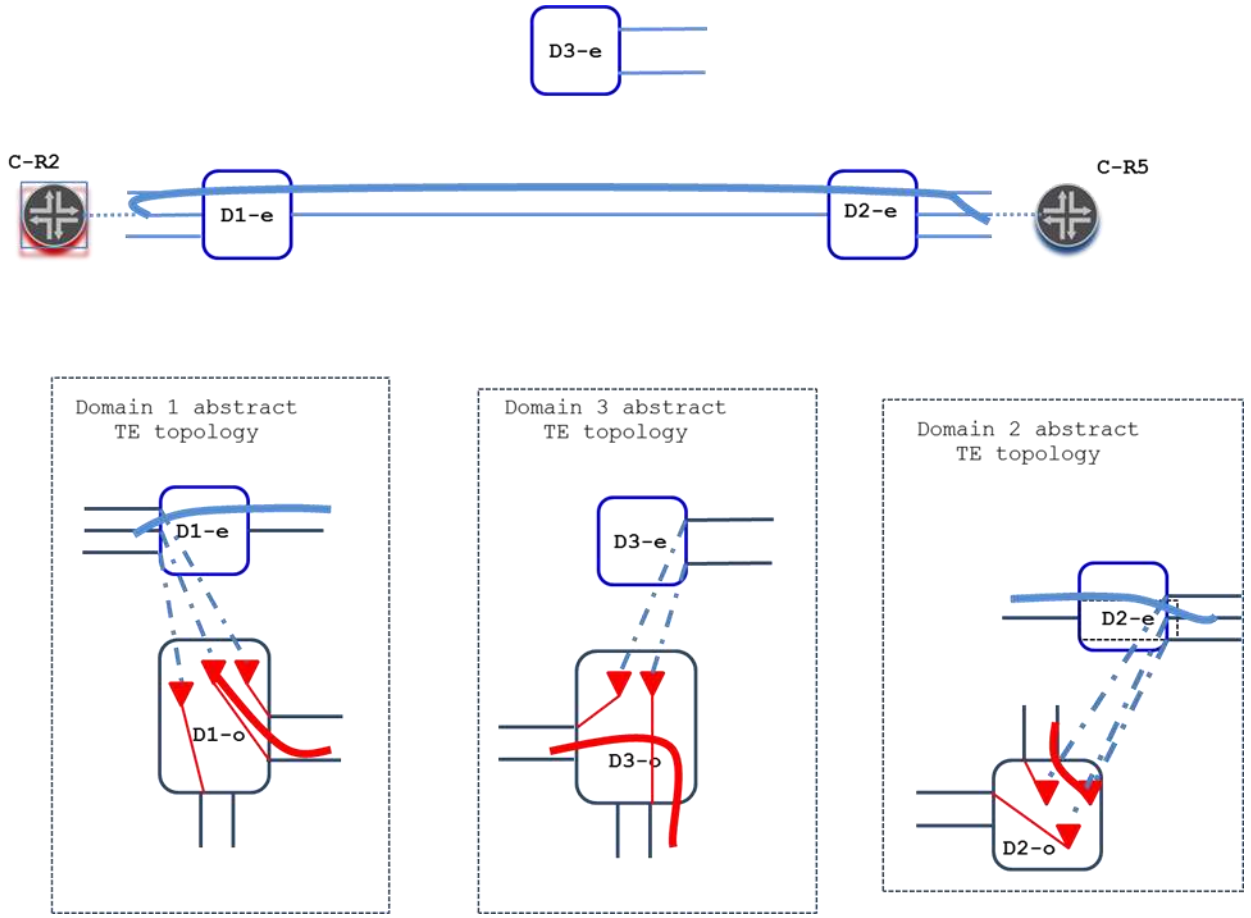

Figure 38. A new Ethernet layer TE link supported by ODUk layer TE tunnel is added to the provided and merged abstract TE topologies

IF all involved DCs confirm successful setup completion, the requested transport service, as well as the supporting server layer hierarchy TE tunnel, will be provisioned as depicted in Figure 39. If one of the DCs fails to set up its segment in either of the layers, all successfully provisioned segments will be requested by HC to be released.

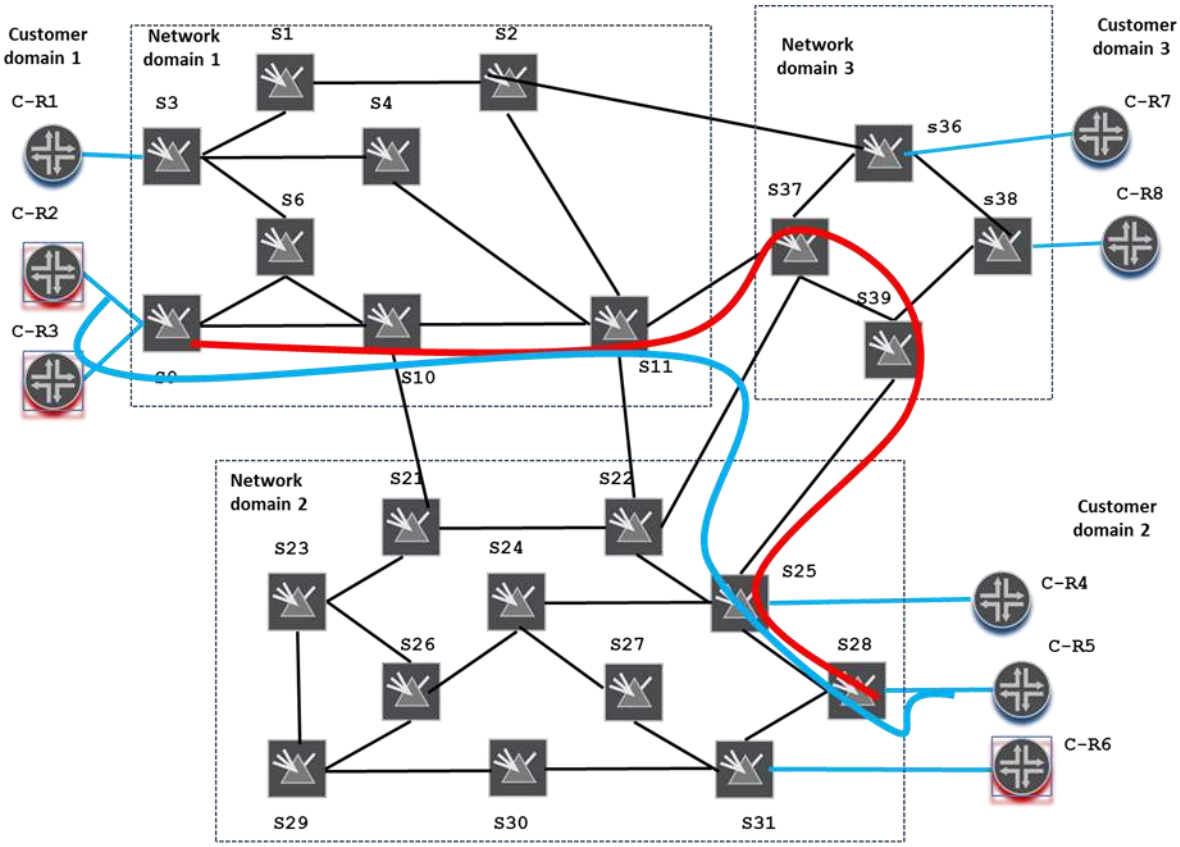

Figure 39. Ethernet transport service and supporting ODUk TE tunnel are provisioned

4) Transport service teardown coordination

First, HC issues to DC1 and DC2 a configuration request to release the Ethernet layer transport service in the respective domains. After that, all three DCs are requested to release the segments of the supporting ODUk layer hierarchy TE tunnel. While processing the request DC1 and DC2 also remove the dynamic Ethernet layer TE links supported by the respective hierarchy TE tunnel's segments, thus the

network's abstract TE topologies are reverted back to the state as shown in Figures 35 and 36.

2.4. Use Case 4. Transport service control on a ODUk/Och multi-domain transport network with multi-function access links

Configuration (Figure 40): the same as in use case 3, except that all access links in this use case are multi-function links (depicted in the Figure as blue compound lines). Let's assume that, depending on configuration, the multi-function access links in this use case can carry either Ethernet or SDH/STM16 layer payload.

Objective: Set up//delete an unprotected shortest delay SDH/STM16 layer transport service interconnecting C-R2 and C-R5

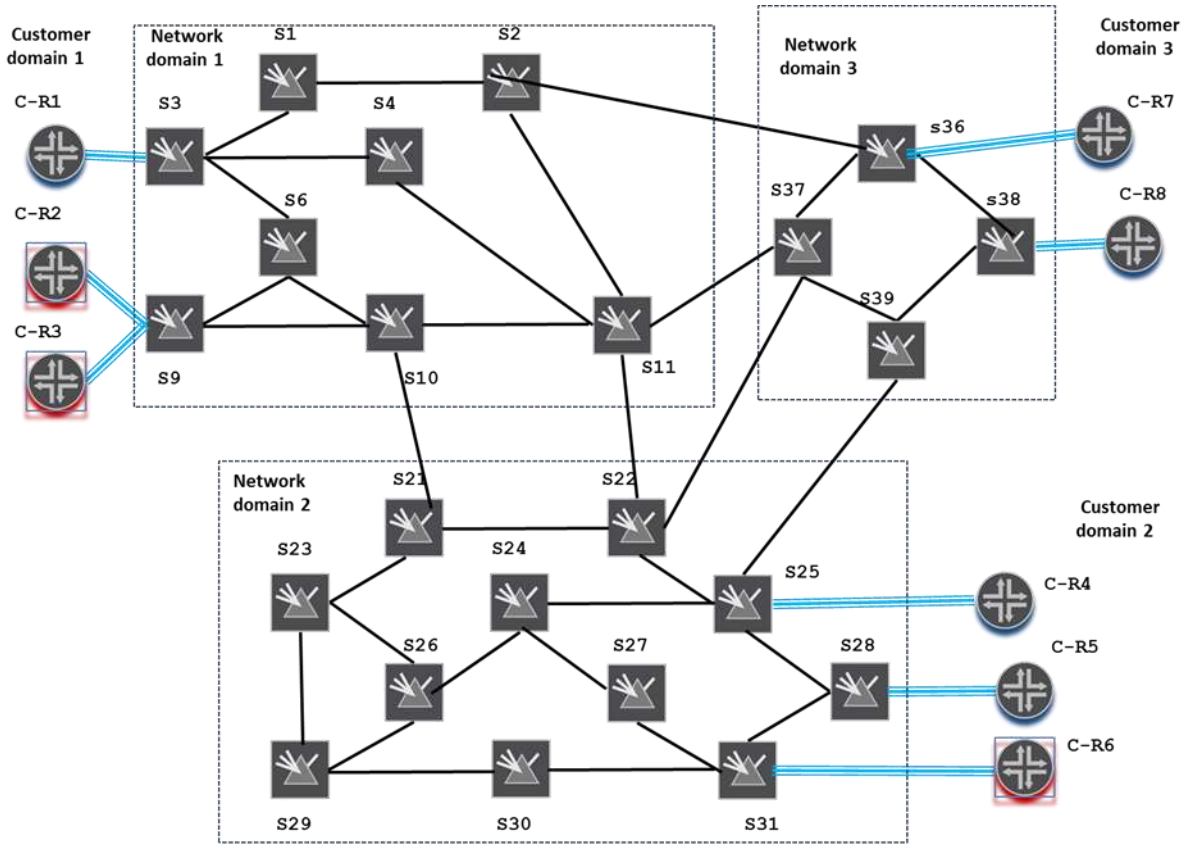

Figure 40. Three-domain ODUk/Och transport network with multi-function access links

1) TE Topology discovery

The TE Topology model considers multi-function links as parallel mutually exclusive TE links each belonging to a separate layer network. For this use case each DC exposes to HC three (ODUk-, Ethernet- and SDH/STM16-layer) abstract TE topologies (generally speaking, one abstract TE topology per each layer network supported by at least one access or inter-domain link). Like in use case 3, the lower layer (ODUk) TE nodes announce hosted by them TE tunnel termination points, TTPs, capable in this case of adopting Ethernet, SDH/STM16 or both layer payloads, The TTPs are attributed with TE inter-layer locks matching ones specified for Ethernet and/or SDH/STM16 TE links.

Ethernet, SDH/STM16 and ODUk layer single-node abstract TE topologies catered to HC by each of the DCs are presented in Figure 41.

HC merges the provided topologies into its own native TE Topology (the merging procedures are described in 1.4). As in use case 3 HC locks the provided TE topologies not only horizontally (i.e. between domains), but vertically (between layers) as well, thus producing a three-layer TE topology homogenously describing the three layers of the entire transport network, as well as the client-server layer adaptation relationships between the layers. This makes the merged TE topology suitable for multi-layer/inter-layer multi-domain transport service path computations. The merged TE topology is presented in Figure 42.

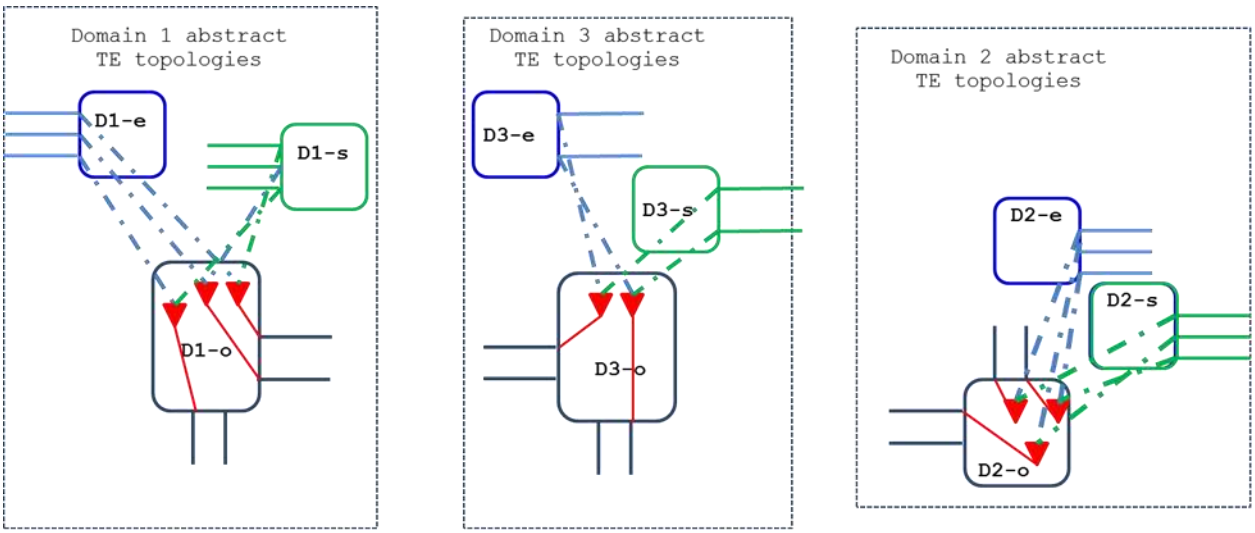

Figure 41. ODUk, Ethernet and SDH/STM16 layer abstract TE topologies exposed by DCs

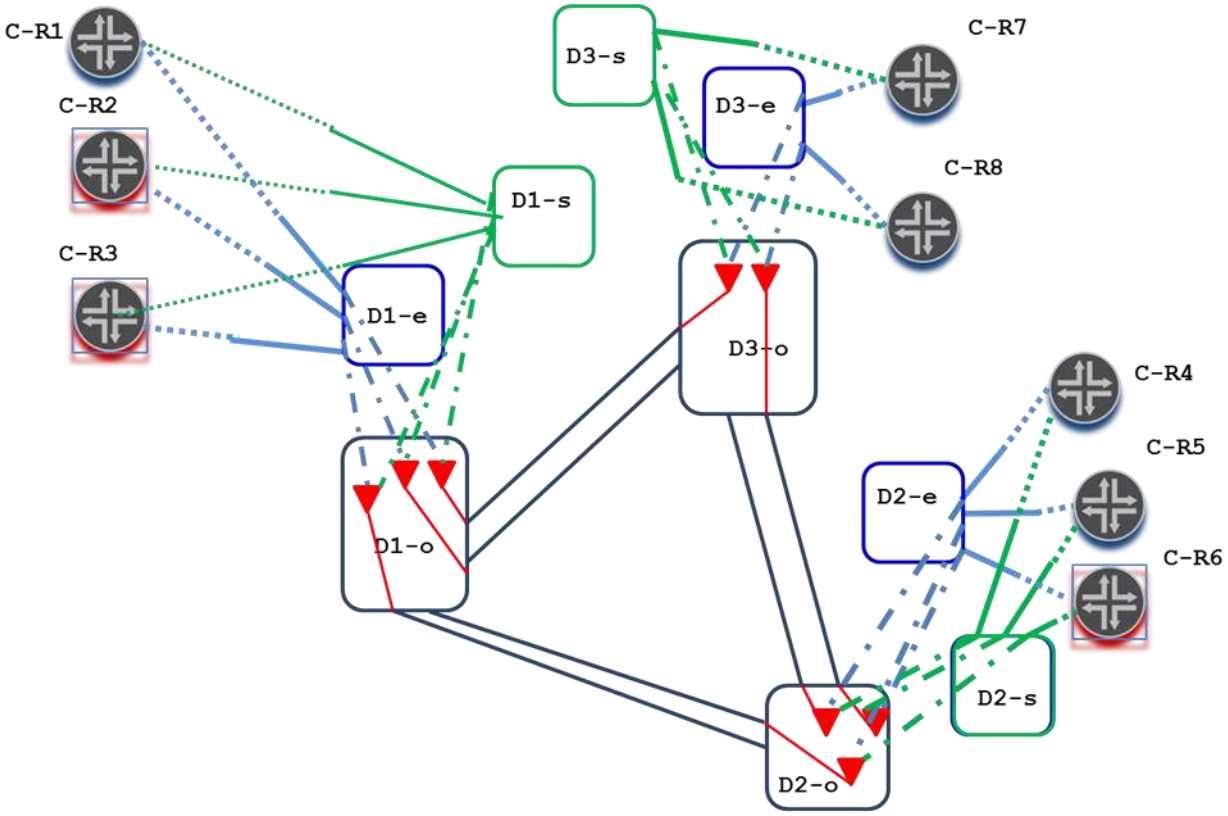

Figure 42. Three-layer three-domain transport network abstract TE topology

2) Transport service path computation

Using the merged TE topology (Figure 42) HC's path computer selects a TE path for the requested transport service. For example, for the SDH/STM16 layer unprotected transport service the resulting TE path could be determined as marked in Figure 43.

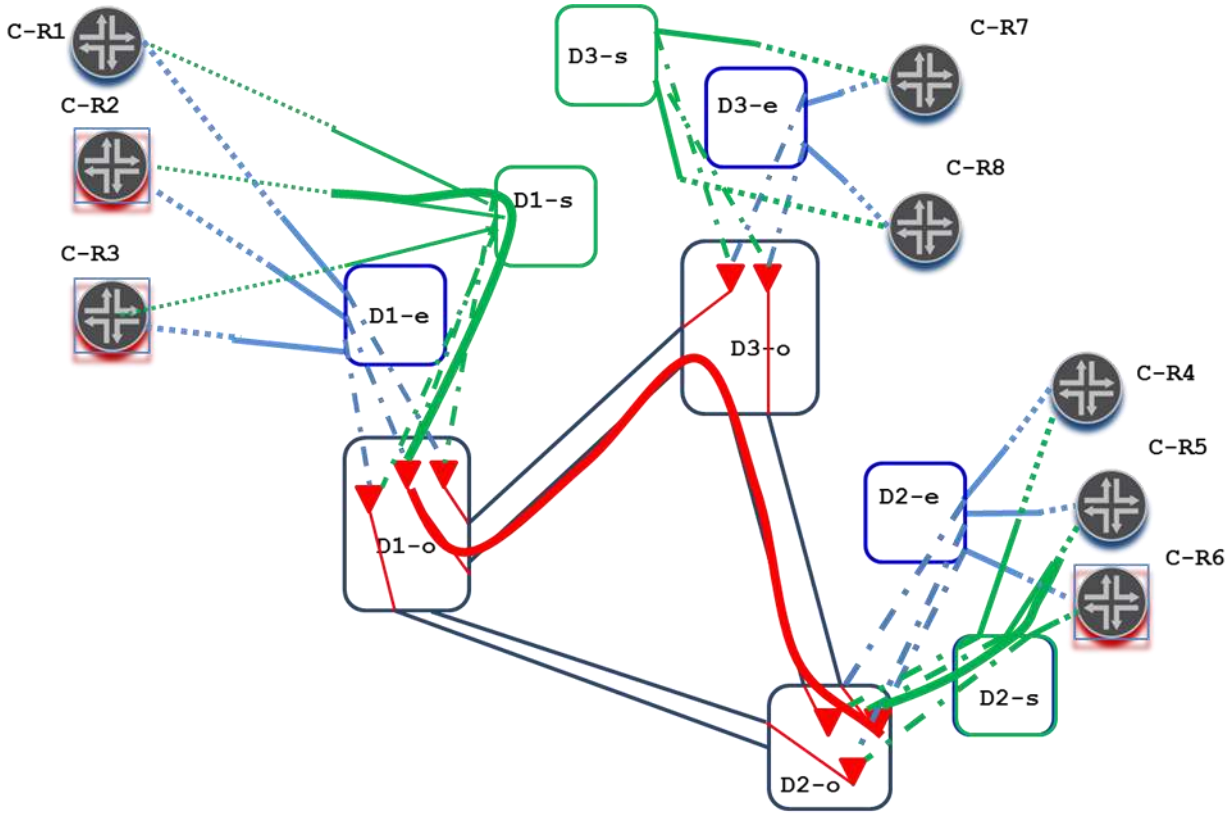

Figure 43. Multi-layer TE path computed for SDH/STM16 layer transport service

3) Transport service setup coordination

Same as in use case 3.

4) Transport service teardown coordination

Same as in use case 3.

2.5. Use Case 5. Real time updates of IP/MPLS layer TE link attributes that depend on supporting transport connectivity (e.g. transport SRLGs, propagation delay, etc.)

Configuration (Figure 26): the same as in use case 1,

Objective: A transport service interconnecting transport ports of two IP routers across a transport network is likely to serve a link in IP/MPLS layer network, which is usually controlled by a client of the

transport network, such as IP/MPLS Controller. Performance of TE applications (e.g. path computer) running on the IP/MPLS Controller depends on the accuracy of IP/MPLS layer TE link attributes. Some of these attributes can change over time and are known real-time only to a transport network controller, such as HC. Examples of said attributes are transport SRLGs, propagation delay metric, protection capacities and status, etc. The objective of this use case is to ensure up-to-date state of said attributes in the IP/MPLS Controller's internal TED via necessary updates provided in a timely manner by the controller (e.g. HC) managing transport connectivity supporting IP/MPLS layer links.

Realization:

- HC exposes and supports IETF TE Topology and TE Tunnel model based NBIs (the same NBIs that are exposed by DCs serving HC);
- IP/MPLS Controller makes use of the exposed NBIs to set up the respective client-provider relationships with HC;
- IP/MPLS Controller uses the TE Tunnel NBI to configure with HC a transport service interconnecting transport ports of a pair of IP routers desired to be adjacent in the IP/MPLS layer network. The TE Tunnel model allows for specifying in the transport service configuration request the TE topology and link IDs of the IP/MPLS TE link the requested transport service will be serving;
- IP/MPLS Controller uses the TE Topology NBI to subscribe with HC on the IP/MPLS TE link notifications with respect to changes in the TE link's attributes, such as SRLGs, propagation delay, protection capabilities/status, etc.;
- HC uses the TE Topology NBI to convey the requested notifications when HC learns the attributes IP/MPLS has expressed interest in or detects any changes since previous notifications (for example, due to network failure restoration/reversion procedures happened to the transport connectivity that supports the failure affected IP/MPLS links)

2.6. Use Case 6. Virtual Network Service

Configuration (Figure 26): the same as in use case 1,

Objective: Set up two Virtual Networks for the client, with Virtual Network 1 interconnecting customer IP routers C-R1, C-R7 and C-R4 over a single-node abstract TE topology, and Virtual Network 2

interconnecting customer IP routers C-R2, C-R3, C-R8, C-R5 and C-R6 over a full mesh link abstract TE topology as depicted in Figure 44.

[Note: A client of a transport network may want to limit the transport network connectivity of a particular type and quality within distinct subsets of its network elements interconnected across the transport network. Furthermore, a given transport network may serve more than one client. In this case some or all clients may want to ensure the availability of transport network resources in case dynamic (re-)connecting of their network elements across the transport network is envisioned. In all such cases a client may want to set up one or more Virtual Networks over provided transport network]

1) Virtual Network setup

From the client's point of view a Virtual Network setup includes the following procedures:

- Identifying the Virtual Network membership – a subset of the client's network elements/ports to be interconnected over the abstract TE topology configured for the Virtual Network. Note that from the transport network provider's point of view this effectively determines the list of abstract TE topology's open-ended access TE links;

- Deciding on the Virtual Network's abstract TE topology type (e.g. single-node vs. link mesh), optimization criterion (e.g. shortest delay vs. smallest cost), bandwidth, link disjointedness, adaptation capabilities and other requirements/constraints, as well as, whether the TE tunnels supporting the abstract TE topology need to be pre-established or established on demand (i.e. when respective abstract TE topology elements are selected for a client transport service);

- Using the IETF TE Topology model based NBI exposed by the transport network controller (i.e. HC), configure the Virtual Network's abstract TE topology. Let's assume that in this use case the abstract TE topology for Virtual Network 1 is configured as a single-node abstract TE topology (see section 1.3.1) with the abstract TE node's detailed connectivity matrix optimized according to the shortest delay criteria. Likewise, the abstract TE topology for Virtual Network 2 is configured as a full-mesh link abstract TE topology (see section 1.3.2) optimized according to the smallest cost criteria with each of the abstract TE links to be supported by pre-established end-to-end protected TE tunnels.

[Note: Virtual Network's abstract TE topology (re-)configuration/negotiation process is no different from one that happens, for example, between HC and its providers, DCs, and is described in section 1.5]

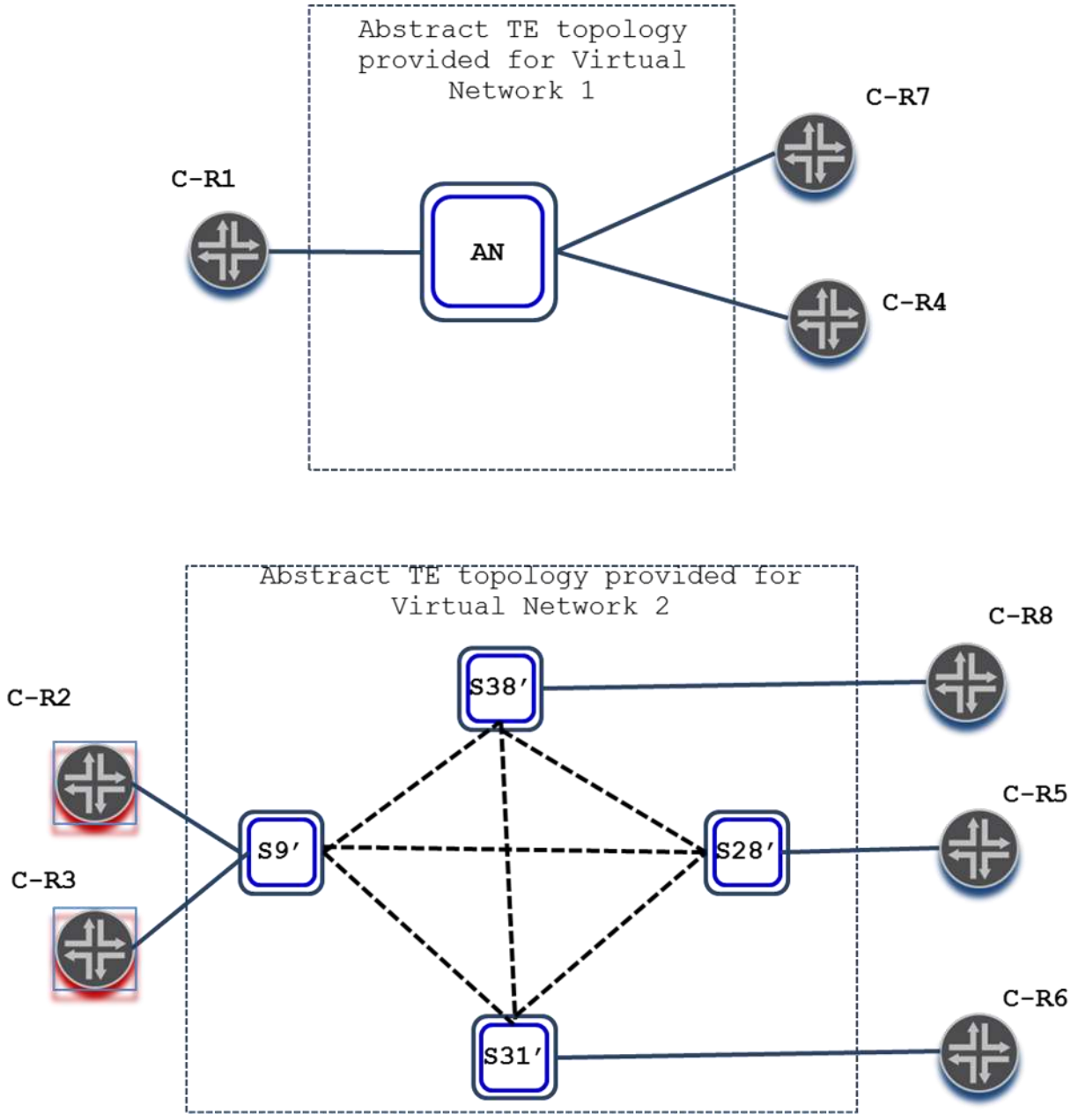

Figure 44. Virtual Networks provided for a transport network client

2) Using Virtual Network

Recall that use case 1 was about setting up a transport service interconnecting customer network elements C-R2 and C-R5 across the transport network. With the Virtual Network 2 in place, the client could have used the Virtual Network's TE topology to select a TE path for the service. The TE Tunnel model based NBI allows for the client to specify the Virtual Network's TE topology ID, as well, as the selected TE path (for example, as marked in Figure 45) as a configured path attribute in the transport service configuration request to ensure that the intended transport network resources are used for the service.

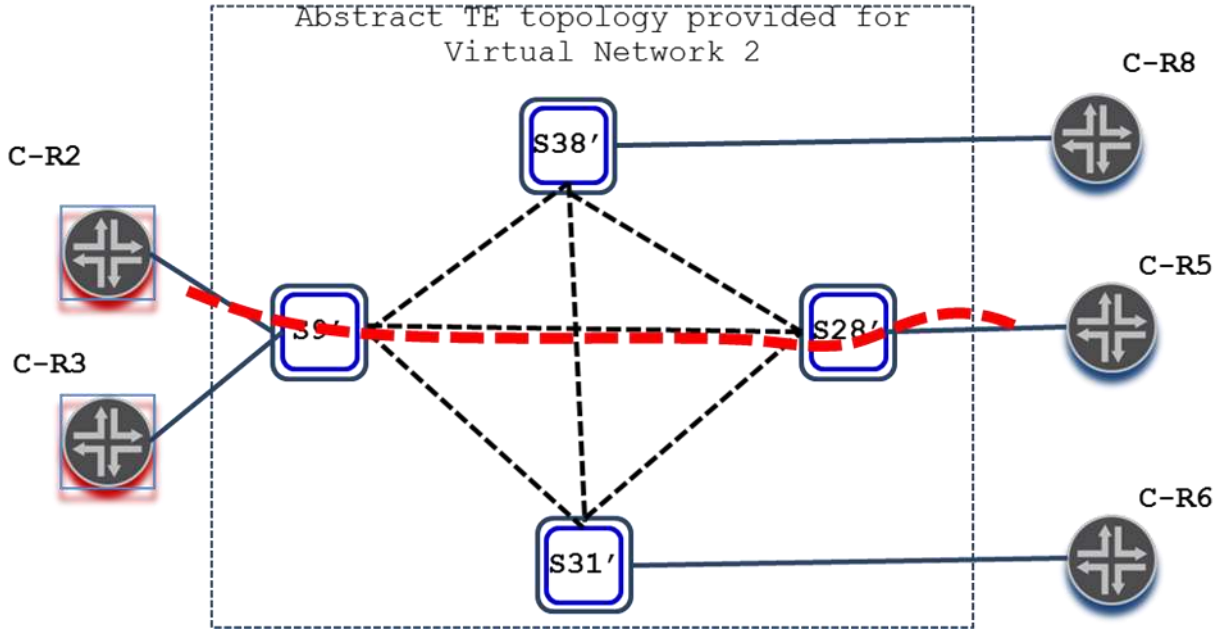

Figure 45. Transport service TE path is selected on Virtual Network's TE topology

3. Security Considerations

This document does not define networking protocols and data, hence are not directly responsible for security risks.

4. IANA Considerations

This document has no actions for IANA.

5. References

5.1. Normative References

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

[I-D.ietf-teas-yang-te-topo] Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for TE Topologies", draft-ietf-teas-yang-te-topo-08 (work in progress), March 2017.

[I-D.ietf-teas-yang-te] Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-06 (work in progress), March 2017.

5.2. Informative References

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

6. Acknowledgments

TBD.

Appendix A. Data Examples

This section contains examples of an instance data in the JSON encoding [RFC7951].

A.1. Use Case 1

In the use case described in Section 2.1. , there are three provider network domains, each of them is represented as an abstract TE topology. The JSON encoded example data configurations for the three domains are:

A.1.1. Domain 1

```
{
  "networks": {
    "network": [
      {
        "network-types": {
          "te-topology": {}
        },
        "network-id": "otn-domain1-abs",
        "provider-id": 201,
        "client-id": 300,
        "te-topology-id": "te-topology:otn-domain1-abs",
        "node": [
          {
            "node-id": "D1",
            "te-node-id": "2.0.1.1",
            "te": {
              "config": {
                "te-node-attributes": {
                  "domain-id" : 1,
                  "is-abstract": [null],
                  "underlay-topology": "domain1-och",
                  "connectivity-matrices": {
                    "is-allowed": true,
                    "max-link-bandwidth": "1,2",
                    "te-default-metric": 10,
                    "connectivity-matrix": [
                      {
                        "id": 10302,
                        "from": "1-0-3",
```

```
          "to": "1-2-0"
        },
        {
          "id": 10203,
          "from": "1-0-2",
          "to": "1-3-0"
        },
        {
          "id": 10311,
          "from": "1-0-3",
          "to": "1-11-0"
        },
        {
          "id": 11103,
          "from": "1-0-11",
          "to": "1-3-0"
        },
        {
          "id": 10903,
          "from": "1-0-9",
          "to": "1-3-0"
        },
        {
          "id": 10309,
          "from": "1-0-3",
          "to": "1-9-0"
        },
        {
          "id": 10910,
          "from": "1-0-9",
          "to": "1-10-0"
        },
        {
          "id": 11009,
          "from": "1-0-10",
          "to": "1-9-0"
        },
        {
          "id": 20910,
          "from": "1-1-9",
          "to": "1-10-0"
```

```
                        },
                        {
                          "id": 21009,
                          "from": "1-0-10",
                          "to": "1-9-1"
                        },
                        {
                          "id": 20911,
                          "from": "1-1-9",
                          "to": "1-11-0"
                        },
                        {
                          "id": 21109,
                          "from": "1-0-11",
                          "to": "1-9-1"
                        }
                      ]
                    }
                  }
                },
                "termination-point": [
                  {
                    "tp-id": "1-0-3",
                    "te-tp-id": 10003
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  },
                  {
                    "tp-id": "1-3-0",
                    "te-tp-id": 10300
                    "te": {
                      "config": {
```

```
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-0-9",
                "te-tp-id": 10009
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-9-0",
                "te-tp-id": 10900
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-1-9",
                "te-tp-id": 10109
                "te": {
```

```
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-9-1",
                  "te-tp-id": 10901
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-2",
                  "te-tp-id": 10002
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-2-0",
                  "te-tp-id": 10200
```

```
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-10",
                  "te-tp-id": 10010
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-10-0",
                  "te-tp-id": 11000
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-11",
```

```
                  "te-tp-id": 10011
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-11-0",
                  "te-tp-id": 11100
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-1-11",
                  "te-tp-id": 10111
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
```

```
                    "tp-id": "1-11-1",
                    "te-tp-id": 11101
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  }
                ]
              }
            ]
          }
        ]
      }
    }
```

A.1.2. Domain 2

```
    {
      "networks": {
        "network": [
          {
            "network-types": {
              "te-topology": {}
            },
            "network-id": "otn-domain2-abs",
            "provider-id": 202,
            "client-id": 300,
            "te-topology-id": "te-topology:otn-domain2-abs",
            "node": [
              {
                "node-id": "D2",
                "te-node-id": "2.0.2.2",
                "te": {
                  "config": {
                    "te-node-attributes": {
```

```
"is-abstract": [null],
"underlay-topology": "domain2-och",
"connectivity-matrices": {
  "is-allowed": true,
  "max-link-bandwidth": "1,2",
  "te-default-metric": 10,
  "connectivity-matrix": [
    {
      "id": 12125,
      "from": "1-0-21",
      "to": "1-25-0"
    },
    {
      "id": 12521,
      "from": "1-0-25",
      "to": "1-21-0"
    },
    {
      "id": 12128,
      "from": "1-0-21",
      "to": "1-28-0"
    },
    {
      "id": 12821,
      "from": "1-0-28",
      "to": "1-21-0"
    },
    {
      "id": 12231,
      "from": "1-0-22",
      "to": "1-31-0"
    },
    {
      "id": 13122,
      "from": "1-0-31",
      "to": "1-22-0"
    },
    {
      "id": 22228,
      "from": "1-1-22",
      "to": "1-28-0"
```

```
                        },
                        {
                          "id": 22822,
                          "from": "1-0-28",
                          "to": "1-22-1"
                        },
                        {
                          "id": 12528,
                          "from": "1-0-25",
                          "to": "1-28-0"
                        },
                        {
                          "id": 12825,
                          "from": "1-0-28",
                          "to": "1-25-0"
                        }
                      ]
                    }
                  }
                },
                "termination-point": [
                  {
                    "tp-id": "1-0-21",
                    "te-tp-id": 10021
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  },
                  {
                    "tp-id": "1-21-0",
                    "te-tp-id": 12100
                    "te": {
                      "config": {
```

```
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-0-22",
                "te-tp-id": 10022
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-22-0",
                "te-tp-id": 12200
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-1-22",
                "te-tp-id": 10122
                "te": {
```

```
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-22-1",
                  "te-tp-id": 12201
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-25",
                  "te-tp-id": 10025
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-25-0",
                  "te-tp-id": 12500
```

```
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-1-25",
                  "te-tp-id": 10125
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-25-1",
                  "te-tp-id": 12501
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-28",
```

```
                  "te-tp-id": 10028
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-28-0",
                  "te-tp-id": 12800
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-31",
                  "te-tp-id": 10031
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
```

```
                    "tp-id": "1-31-0",
                    "te-tp-id": 13100
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  }
                ]
              }
            ]
          }
        ]
      }
    }
```

A.1.3. Domain 3

```
    {
      "networks": {
        "network": [
          {
            "network-types": {
              "te-topology": {}
            },
            "network-id": "otn-domain3-abs",
            "provider-id": 203,
            "client-id": 300,
            "te-topology-id": "te-topology:otn-domain3-abs",
            "node": [
              {
                "node-id": "D3",
                "te-node-id": "2.0.3.3",
                "te": {
                  "config": {
                    "te-node-attributes": {
```

```
"is-abstract": [null],
"underlay-topology": "domain3-och",
"connectivity-matrices": {
  "is-allowed": true,
  "max-link-bandwidth": "1,2",
  "te-default-metric": 10,
  "connectivity-matrix": [
    {
      "id": 13638,
      "from": "1-0-38",
      "to": "1-38-0"
    },
    {
      "id": 13836,
      "from": "1-0-38",
      "to": "1-36-0"
    },
    {
      "id": 13639,
      "from": "1-0-36",
      "to": "1-39-0"
    },
    {
      "id": 13936,
      "from": "1-0-39",
      "to": "1-36-0"
    },
    {
      "id": 23636,
      "from": "1-0-36",
      "to": "1-36-1"
    },
    {
      "id": 33636,
      "from": "1-1-36",
      "to": "1-36-0"
    },
    {
      "id": 13739,
      "from": "1-0-37",
      "to": "1-39-0"
```

```
                        },
                        {
                          "id": 13937,
                          "from": "1-0-39",
                          "to": "1-37-0"
                        },
                        {
                          "id": 23737,
                          "from": "1-0-37",
                          "to": "1-37-1"
                        },
                        {
                          "id": 33737,
                          "from": "1-1-37",
                          "to": "1-37-0"
                        }
                      ]
                    }
                  }
                },
                "termination-point": [
                  {
                    "tp-id": "1-0-36",
                    "te-tp-id": 10036
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  },
                  {
                    "tp-id": "1-36-0",
                    "te-tp-id": 13600
                    "te": {
                      "config": {
```

```
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-0-37",
                "te-tp-id": 10037
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-37-0",
                "te-tp-id": 13700
                "te": {
                  "config": {
                    "interface-switching-capability": [
                      {
                        "switching-capability": "switching-otn",
                        "encoding": "lsp-encoding-oduk"
                      }
                    ],
                  }
                }
              },
              {
                "tp-id": "1-1-37",
                "te-tp-id": 10137
                "te": {
```

```
          "config": {
            "interface-switching-capability": [
              {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
              }
            ],
          }
        }
      },
      {
        "tp-id": "1-37-1",
        "te-tp-id": 13701
        "te": {
          "config": {
            "interface-switching-capability": [
              {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
              }
            ],
          }
        }
      },
      {
        "tp-id": "1-0-39",
        "te-tp-id": 10039
        "te": {
          "config": {
            "interface-switching-capability": [
              {
                "switching-capability": "switching-otn",
                "encoding": "lsp-encoding-oduk"
              }
            ],
          }
        }
      },
      {
        "tp-id": "1-39-0",
        "te-tp-id": 13900
```

```
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-36",
                  "te-tp-id": 10036
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-36-0",
                  "te-tp-id": 13600
                  "te": {
                    "config": {
                      "interface-switching-capability": [
                        {
                          "switching-capability": "switching-otn",
                          "encoding": "lsp-encoding-oduk"
                        }
                      ],
                    }
                  }
                },
                {
                  "tp-id": "1-0-38",
```

```
                    "te-tp-id": 10038
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  },
                  {
                    "tp-id": "1-38-0",
                    "te-tp-id": 13800
                    "te": {
                      "config": {
                        "interface-switching-capability": [
                          {
                            "switching-capability": "switching-otn",
                            "encoding": "lsp-encoding-oduk"
                          }
                        ],
                      }
                    }
                  }
                ]
              }
            ]
          }
        ]
      }
    }
```

Authors' Addresses

Igor Bryskin
Huawei Technologies
Email: Igor.Bryskin@huawei.com

Vishnu Pavan Beeram
Juniper Networks

Email: vbeeram@juniper.net

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

Xufeng Liu
Jabil
Email: Xufeng_Liu@jabil.com