Multi-layer Multi-domain Network Topology Abstractions Using ONF Transport API

Karthik Sethuraman, NEC

September 2019

*animated slides
ONF Transport API (TAPI): Functional Architecture

Industry Adoption: Functional Interface solution that best fulfills conflicting requirements of stability (future proof, interoperability) & flexibility/agility (technology evolution).

Shared Network Information Context

Transport API or Other NBIs

Transport API or Other SBIs

Network Elements

SDN Controller

Application

SDN Controller

Topology Service

Connectivity Service

OAM Service

Path Computation Service

Virtual Network Service

Equipment Inventory Service

Notification Service
MEF: Lifecycle Service Orchestration Reference Architecture (LSO RA)

Customer Domain

- Customer Application Coordinator

SP Domain

- Business Applications
- Service Orchestration Functionality
  - CANTATA (CUS:BUS)
  - ALLEGRO (CUS:SOF)
  - LEGATO (BUS:SOF)
  - PRESTO (SOF:ICM)
- Infrastructure Control and Management
  - ADAGIO (ICM:ECM)
- Element Control and Management
  - INTERLUDE (SOF:SOF)
  - SONATA (BUS:BUS)
  - PRESTO (SOF:ICM)

Partner Domain

- Business Applications
- Service Orchestration Functionality
- Infrastructure Control and Management
- Element Control and Management

Diagram Notes:
- CUS: Customer Application Coordinator
- BUS: Business Applications
- SOF: Service Orchestration Functionality
- ICM: Infrastructure Control and Management
- ECM: Element Control and Management

ENN I
Simple Physical Network Example to illustrate T-API

- A Network Provider (Blue) with two Customers (Red and Green)
- All UNI interfaces are ETH (e.g. 10GE), I-NNI interfaces are OTU (e.g. 100G OTN)
- All PE-NE are ODU/ETH switch capable, while P-NE is only ODU switch capable

CE – Customer Edge
PE – Provider Edge
P - Provider

Logical Termination Points
- Service Interface Point
- Node Edge Point (Network Edge)
- Node Edge Point (Network Internal)
T-API Contexts for the Simple Network Example

(based on ONF Architecture v1.1)
Example Topology Abstractions in the Shared Context

Topology

Node

Node Edge Point (NW Edge)

Node Edge Point (NW Internal)

Service Interface Point

Link

Mapping

Transitionnal Link

Pink Admin Context

Red Shared Context

Green Shared Context
Client-1 (Red) Shared Context: Single Node Topology

- Single Node abstraction example
- *Node* and its *NodeEdgePoints* provide some approximation of the network capabilities
- *ConnectivityService* can be requested between *ServiceInterfacePoints*
- *Connections* appear as cross-connections across node, no visibility of underlying route
Client 2 (Green) Shared Context: Multi-Node Topology

- Multiple Nodes (PEs) Topology example
- Node and its NodeEdgePoints provide reasonable information of their capabilities
- ConnectivityService can be requested between ServiceInterfacePoints
- Top-level Connection is recursively decomposed into lower-level Connections, 1 per Node
- Connection route can be traced over the exposed Topology
Admin (Pink) Shared Context: Multi-layer Topology

- Each physical device is represented by a separate *Node* per supported layer (ETH & ODU)
- *Node* and its *NodeEdgePoints* provide information of their capabilities at that layer
- *Transitional Links* interconnect the *NodeEdgePoints* at different layers
- Top-level *Connection* is recursively decomposed into lower-level *Connections*, 1 per Node
- Top-level *Connections* at lower (server) layer result in *Links* at upper (client) layer
Recursive Node & Topology aspects of Forwarding Domain

- **Node aspect of FD**
- **Topology aspect of FD**
- **Observer**
- **FD (Node)**
- **FD (Topology)**

**TAPI Context**

- Context appears as a Topology N of one Node A and SIPS (off-network relationships/Links)
- Node-A appears a Topology of Nodes B & C and Link B-C
- Node-C appears a Topology of Nodes C.1-C.4 & Links between them
Connection C appears as a route of Connections C.1, C.3, C.4 & in-between Links

Connection A appears a route of Connections B, C and Link B-C

Top-level “Network” Connection A
TAPI 2.2 Example 1: Single-level Topology, Network-Node (Single) abstraction

- Single Node abstraction – for simple TAPI applications/clients that does not want to concern itself with Network topology & routing
- Ability to expose top-level “Network” Connection only – No way to represent top-level Connection’s route or its lower-level decomposed “cross” connections
TAPI 2.2 Example 2a: Single-level Topology, Network abstraction w/ implicit Node

- Implicit Top-level Single Node (and its encapsulated Topology)
- Provides ability to expose top-level “Network” Connection and its Connection-topology/route as well as its decomposed lower-level “Cross” Connections
- Top-level Connection is NOT bounded by an explicit Node, while lower-level connections have a bounding Node
TAPI 2.2 Example 2b: Single-level Topology, Network abstraction /w multi-level implicit Nodes

- A variation of example 2a – but with multiple levels of implicit Nodes (and their encapsulated Topology hierarchy)
- Provides ability to expose top-level “Network” Connection and its Connection-topology/route as well as multiple levels of Connection decomposition
- Top-level Connection as well as intermediate-level connections are NOT bounded by an explicit Node, while lowest-level connections have a bounding Node
TAPI 2.2 Example 3: 2-level Topology, Network abstraction w/ Explicit Node

- This abstraction emerges from example 2a, with distinction of an Explicit bounding Node for “Network” Connection which allows for clean representation of forwarding capability prior to setting up of connectivity
- Provides ability to expose top-level “Network” Connection and its Connection-topology/route as well as its decomposed lower-level “Cross” Connections
- All Connections are bounded by an explicit Node
TAPI 2.2 Example 4: Multi-level Topology Partitioning abstraction

- Emerges from example 2b - with distinction of multiple levels of Explicit Nodes (and their encapsulated Topology hierarchy)
- Provides ability to expose multi-level Connections, each bounded by an explicit Node
TAPI 2.2 v/s RFC8345 Topology Models: Simplified View

TAPI 2.2 Topology (Partitioning & Layering)  RFC 8345 Topology (View mapping)
TAPI 3.0 augments RFC8345 Topology Model (under discussion)

Tapi 2.2 Topology (Partitioning & Layering)  RFC 8345 Topology (View mapping)
TAPI 3.0 Example 5: Multi-level Topology Partitioning via Mapping abstraction

- Emerges from example 4 - with distinction of multiple clones NEPs & CEPs at every hierarchy level
TAPI 3.0 Example 6: Multiple Topology Views via Mapping abstraction

- Provides ability to provide & map different intermediate views of a single underlying Topology.

Node Topology

Network Topology

Topology C

Topology B

Topology A

TAPI Context
ONF ODTN (Open Disaggregated Transport) Architecture

With OLS Controller
Operator Domain Connectivity Service & Resources

Abbreviations
TPD – Transponder Node
RDM – ROADM Node
UNI – User-Network Interface
NNI – Network-Network Interface
DSR – Digital Signal Rate
OTSi – Optical Tributary Signal
OTSiA – OTSi Assembly
MC – Media Channel

Logical Termination Points shown
- Service Interface Point
- Node Edge Point (Network Edge)
- Node Edge Point (Network Internal)
- Connectivity Service End Point
- Connection / Connection End Point
- Photonic Connection
- Photonic Media Channel
Thank You

Follow Up Links:
https://wiki.opennetworking.org/display/OTCC/TAPI