Open Network API Development

Lyndon Ong (lyong@ciena.com)
Project Lead, ONF Open Transport Config & Control

Nigel Davis (ndavis@ciena.com)
Project Co-Lead, ONF Open Information Modeling and Tooling

March 27, 2018
Ciena’s Vision for the Adaptive Network

5G Networks
- Enhanced Mobile Broadband
  - Massive Capacity
  - Video, VR
- IoT
  - Massive Connectivity
  - SensorNets, Drones
- URLL
  - Latency & Reliability
  - Factory, Car Automation

Goal: move policies & data between systems in a Multi-Domain, Multi-Vendor network
- Open, Standard APIs
Drivers: Vision and apparent/necessary trends

Vision:
- The Adaptive Network
  - To deal with pace, scale, complexity and need for efficiency/consistency

Trend:
- To coding for patterns
  - From coding per case
- To continuums
  - From distinct layers and silos
- To assemblies of versatile focussed components
  - From branded monolithic “solutions”
- To modular fluid federation of pattern based models
  - From monolithic static conflicting taxonomical hierarchical models
- To an Agile Value Fabric of versatile and continually evolving interacting service provider organizations
  - From fixed role slow change market players
- To evolving open agile standards
  - From boom-bust rigid fragile standards
Evolving Interfaces

OLD
- Reinvention of new models for every new use case
- Focus on Spec development as the end goal
- Use protocol stack vendors to implement and make changes
- Example: RSVP (1997)

NEW
- Model-driven with common base model across use cases
- Focus on model/software development with Specs following
- Automated tools for mapping from model to implementation
- Example: T-API

- One structure (grammar) that is essentially extensible but over time stabilizes
- Infinitely extensible noun set
- Interpretable constraint set (dynamic extension)
- Opportunity for sophisticated expression of delegated interdependent outcome structure
- Per exchange-set encoding choices negotiated and supported by the infrastructure allowing on-the-fly encoding optimization
Principles: Control System View vs. System

As usual... recursive, fractal, abstractions and views all apply.

Key
- Actual running component
- View of component
- Port

ControlSystemView

Through

Controller

To

About

Comms + port

Controller realized by “Controlled Entities” (Components)

Client System Observer

Controlled System

Projection

Projection

Controller System realized by “ControlledEntities” (Components)
Principles: Component vs. System

Any functionality can be considered in terms of a system of components
- Where the system can be viewed as a component

Any view of functionality offered by a component can be considered in terms of a system of components
- The system will usually be an abstraction of the system that “actually” supports the Component

Key
- **Simplified view of a component**
- **Consume(r) Interface**
- **Provide(r) Interface**
- **Functionality**
- **Services**
- **Owned Data**
- **Workflow**
- **Policy**
- **Functional Capability**

[TMF IG11118] Figure 1 The FMO component interface and structural overview
Recursion of control levels
Essentially the same at all levels
Automated management is control
A “layer” of control revolves round a model of the controlled things
A control layer will have a repository reflecting:
  • Intent to support a contract/agreement with or request from a client
  • Expected and current state of the controlled things in the context of the chosen realization

A control “layer” will present, to its clients, in client terminology, an abstracted view of what it controls where that view aligns with the current contract in terms of:
  • Potential for providing capability
  • Actual provided capability
Principles: Outcome-oriented constraint-based interaction

- **Potential Client needs and potential provider capabilities intersect**
  - Both have shared understanding of some semantics in a space where the provider has capabilities and the client needs
  - Both have (or agree) a common representation of the semantics (which may be a mapping from their preferred internal views)
  - The potential provider does not expose all aspects of all capabilities (e.g., “how” may be vague or opaque)
  - The potential client does not expose all aspects of the need (e.g., “why” may be vague or opaque)
- **Potential client and provider negotiate**
  - This process may increase the semantic intersection and may refine the representation
- **An agreement is reached (provider intention and client expectation)**
  - The agreement will be expressed mainly in terms of what will be achieved and not how it will be achieved
    - Where the client and provider have limited semantic overlap the agreement may be in terms of experience (e.g., apparent adjacency experience) where many of the parameters will be in client terms and potentially as a client oriented profile (e.g., performance). The service is the experience (and the experience is the outcome)
    - Where the client and provider have an extensive overlap, e.g., as the provider is sub-contracting, then the agreement may be more in terms of an abstract realization outcome (e.g., an Ethernet Connection). The service is the realizing (delivery) of the connection, the connection is a resource
  - When the agreement has elements of “how”, then these are invariable abstract with respect to the real “how” (it will not be in terms of the transistors…)
  - In all cases the agreement is in terms of achieving some state (e.g., apparent adjacency, connectivity) where that state is specified in terms of constraints
    - It is not necessary to impose the process used to achieve the Outcome/Experience (as this is the responsibility of the provider), so verbs such as “Create” and “Set” are not necessary/relevant
Principles: Model Driven Solution

Build Core Model:
- Model the problem space semantics for transport
- Focuses on long lived stable structures and patterns in each domain of relevance
- Provides a pattern (Specification) for extension to augment structure with specific properties and substructure

Model views:
- Prune and refactor the Core Model to form model views (such as TAPI and WDAPI)
- Support pruning and refactoring with tooling

Technology specifications:
- Add branches for more detailed technology specific properties

Tooling:
- UML for high level construction, using open source Eclipse environment
- Automated mapping to YANG data model, JSON schema, reference implementation
Implementation of the Common Core Model – Transport API (TAPI)

Pruning and refactoring process

- Prune model for narrower, purpose-specific view of the Information Model
  - Prelude to tool-based mapping into protocol-specific schema and encoding
  - Purpose in this case is Transport network configuration and control
- Refactor model for compatibility with well-known prior terminology and structure
  - Transport networks
    - Topology
    - Nodes and Links
    - Connectivity Services

Core Model: [https://www.opennetworking.org/software-defined-standards/models-apis/](https://www.opennetworking.org/software-defined-standards/models-apis/)  - Information Modeling Project
Transport API - Multi-Domain/Multi-Vendor Integration

- Multi-Domain/Multi-Vendor
- Greenfield/Brownfield
- Integrated control

Standard, Interoperable API across controller/technology/vendor types
Group leader: Karthik Sethuraman, NEC
See https://www.opennetworking.org/projects/open-transport/
History of TAPI Work

- **2014-15**
  - OIF defines framework for Transport SDN
  - OIF prototype demonstration of SDN OpenFlow SBI and REST-based NBI
  - ONF Open Transport group initiates Transport API development

- **2016-17:**
  - ONF completes TAPI 1.0 FRD and SDK
  - OIF interop demonstration of T-API 1.0

- **2018:**
  - ONF completes TAPI 2.0 SDK
  - OIF interop demonstration of TAPI 2.0

ONF Open Transport Configuration & Control Project (OTCC)

- Common configuration and control interfaces for transport networks in SDN, defining these interfaces with open source software and software-defined standards.
- [https://www.opennetworking.org/projects/open-transport/](https://www.opennetworking.org/projects/open-transport/)

**Subprojects:**

- Open Transport Information Modeling (Kam Lam, Fiberhome)
- Wireless Transport (Giorgio Cazzaniga, SIAE)
- Transport API (Karthik Sethuraman, NEC)
TAPI Topology in a Nutshell

Node/Link/NEP details:
- Layer protocol
- Capacity
- Admin/Operational State, etc.

Could go down to component level
Key TAPI Features

- **Technology-agnostic API Framework**
  - Standardizes a single core technology-agnostic specification that abstracts common transport network functions

- **Modular & Extensible**
  - Functional features are packaged into small self-contained largely-independent modules
  - T-API Core Spec is designed to be fully extensible
    - Extensions can be Technology, SDO, Operator or Vendor specific

- **Industry-wide Interoperability Objective**
  - ONF Open Source SDN project under Apache 2 license
  - Industry-wide common modeling effort

- **SDK components generated using ONF tools for agile prototyping**
  - YANG schema generated from UML using guidelines developed in an multi-SDO initiative (IISOMI)
  - Swagger/JSON APIs generated from YANG following RESTConf specification
TAPI SDK Development

- **ONF Transport API Functional Requirements – ONF TR-527, June 2016**
  - ONF Open Transport WG Project
  - Input to the T-API SDK (Software Development Kit)
- **Software-wise, TAPI SDK is packaged as multiple Eclipse sub-projects**
  - **Papyrus-UML Information Model**
    - Pruned/refactored version of ONF OIMT Core Info Model
    - Technology-agnostic generic framework + technology specific extensions (OTN, ETH)
  - **YANG Data Schema**
    - auto-generated from UML using ONF OIMT Eagle Tools
  - **Swagger-JSON RestConf API**
    - auto-generated from YANG using ONF OIMT Eagle tools
  - **Reference Implementation (RI) in Python**
- **Iterative design process with code development an integral part of the cycle**
TAPI 2.0 Services

2.0 SDK has been approved and posted on Github site
- UML Model
- YANG Schema and Swagger Specs
- Reference Implementation in python
- https://github.com/OpenNetworkingFoundation/TAPI

TAPI 2.0 vs. 1.0

Topology Service
- Terminology/Model YANG Alignment
- NodeRules – node connectivity constraints
- Link Resilience properties

Connectivity Service
- Terminology/Model YANG Alignment
- Resilience and Topology Constraints in service request
- SwitchControl status to convey protection state

Notification Service
- Subscription and filtering for autonomous event notification
- Support of Alarms and Threshold Crossing Alerts

Technology Spec Models
- Ethernet, OTN ODU and OTSi technology details

OAM Service
- Support for creation and activation/deactivation of Monitoring Points/Sessions

Path Computation and Virtual Network Services
Supporting Cross-Industry Infrastructure

ONF OIMT
ONF Core Information Model
ONF Technology Specification Models

ITU-T SG15
OTN (ITU-T G.874.1)
ETH (ITU-T G.8052)
MPLS-TP (ITU-T G.8132)

NRM
NRP
MEF Models
Packet WAN
Optical Transport
Open CS Implementations

MEF

TAPI SDK (Github)

TAPI FRD
Use cases & Requirements
TAPI UML Information Model
TAPI YANG Data Schema
SWAGGER/REST APIs
Python Reference Implementation
TAPI Open Source Platform Implementation
ODL Nitrogen, ONOS

ONF OTCC TRs

Open Model Profile
UML-YANG Generation Tool
YANG-SWAGGER Generation Tool
Python Stub Generation Tool
EAGLE Modeling Tools

Multi-carrier T-SDN Interop Implementation Agreements

IISOMI – Informal open source project across ETSI NFV, ITU-T, MEF, ONF and TMF model designers

OIF

MEF Models

ONF Technology Specification Models
2016-17 OIF TAPI Interop Testing

Other Labs:
- China Telecom
- China Unicom
- SKT
- Verizon

Other Vendors:
- Fiberhome
- Huawei
- Juniper
- ZTE

Follow-up 2018 TAPI 2.0 Testing in Planning
TAPI-based MEF LSO Development
MEF NRM/NRP TAPI Augmentation

NRP defines several objects that are augmentations of TAPI objects.

NRM defines several classes that are augmentations of TAPI classes.

Other classes to complete the NRM model besides ONF generic model, either locally defined or imported from other standard models.

MEF 59 – 01/18
“Network Resource Management Information Model: Connectivity”

MEF 60 – 01/18
“Network Resource Provisioning Interface Profile Specification”

Contributors:
Amartus
CenturyLink
Ciena
Coriant
Ericsson
Huawei
Infinera
Iometrix
NEC
Nokia
RAD
Design First Approach for APIs

- Topology model defined in UML
- Topology model defined in Swagger Specification
  - Leveraged TAPI open source tools to convert UML -> YANG -> swagger
- Swagger generated server and client templates, as well as test cases
  - Implementation of swagger generated templates.
ONF Wireless Transport Modeling and Testing

**Defining Wireless Transport Information Model**

- Group leader: Giorgio Cazzaniga, SIAE
- Aligned with Core Model for WT specific technology
- Validated by Carrier-driven and hosted Proof of Concept events

<table>
<thead>
<tr>
<th>Sponsor</th>
<th>1st PoC</th>
<th>2nd PoC</th>
<th>3rd PoC</th>
<th>4th PoC</th>
<th>4.1th PoC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sponsor</td>
<td>Telefonica Madrid 10/15</td>
<td>Telefonica/Germany Munich 4/16</td>
<td>AT&amp;T New Jersey 10/16</td>
<td>DT Bonn 6/17</td>
<td>Multilocation 11-12/17</td>
</tr>
<tr>
<td>SDN controller</td>
<td>ONOS</td>
<td>ODL Lithium SR4</td>
<td>ODL Beryllium SR2</td>
<td>ODL Boron SR1</td>
<td>ODL Boron SR3</td>
</tr>
<tr>
<td>MW model</td>
<td>OF port extensions</td>
<td>TR-532 (partial)</td>
<td>TR-532</td>
<td>TR-532</td>
<td>TR-532</td>
</tr>
<tr>
<td>ONF CoreModel</td>
<td>n.a.</td>
<td>CM 1.1</td>
<td>CM 1.1</td>
<td>CM 1.2</td>
<td>CM 1.3</td>
</tr>
<tr>
<td>SBI protocol</td>
<td>OpenFlow</td>
<td>Netconf/YANG</td>
<td>Netconf/YANG</td>
<td>Netconf/YANG</td>
<td>Netconf/YANG</td>
</tr>
<tr>
<td>NBI Integration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ONAP</td>
</tr>
</tbody>
</table>

**4.1 PoC Goals**

- Application of the ONF microwave model (TR-512/532-based) to the RAN access
- Demonstration of a scalable multi-site architecture including multiple SDN controller instances
- Initial integration of the Wireless Transport work into the ONAP architecture
- See [link](https://wiki.opennetworking.org/download/attachments/262144003/4.1th%20Wireless%20Transport%20PoC%20Closing%20meeting?api=v2)
ONF Open Disaggregated Transport Networks

New 2018 project in ONF

• Aimed at implementing an open source solution supporting disaggregation in optical transport
  • Leverages ONOS as the SDN Controller
  • Multi-layer Packet/Optical Integration
  • Phased models for optical disaggregation
    • Open Line System with Transponder and Optical Line System separation initially
• Implementing standard APIs rather than creating new ones
  • TAPI as the Controller NorthBound Interface (NBI)
  • OpenConfig as the SouthBound Interface (SBI) to devices
• Effort led by service providers
  • NTT, Telefonica and others
Migration from Now to Vision

**Refinements and Validation**
- Tooling improvement (moving targets?)
- Refinements to Interaction Flow
  - Expression of needs and constraints
  - Negotiation and error treatment
  - Efficiency and scaling
- Implementation and Testing for Use Case support
  - OIF Interop Testing for 2018, ONF ODTN Testing, MEF PoCs
- Feedback on completeness and consistency

**Enhancements and Expansion**
- Addition of models and detail
  - Media models: ITU-T Optical Media model, Wireless Transport model
  - Device models: mapping to Chassis/Shelf
- Addition of further paths for implementation
  - Additional encodings, mapping paths
References

ONF OIMT wiki https://wiki.opennetworking.org/pages/viewpage.action?pageId=259325962
ONF Core Model https://www.opennetworking.org/software-defined-standards/models-apis/
ONF OTCC wiki https://wiki.opennetworking.org/pages/viewpage.action?pageId=259325960
ONF T-API SDK https://github.com/OpenNetworkingFoundation/TAPI
T-API 2.0 Overview https://github.com/OpenNetworkingFoundation/TAPI/blob/develop/DOCS/TAPI%20WP_Final.docx
ONF ODTN https://www.opennetworking.org/solutions/odtn/