Experimental Evaluation of Control and Monitoring Protocols for Optical SDN Networks and Equipment [Invited Tutorial]

RICARD VILALTA[1,*], CARLOS MANSO[1], NOBORU YOSHIKANE[2], RAMON CASELLAS[1], RICARDO MARTÍNEZ[1], TAKEHIRO TSURITANI[2], ITSURO MORITA[2], AND RAUL MUÑOZ[1]

[1] *Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Castelldefels, Spain*
[2] *KDDI Research Inc., Saitama, Japan*
[*] *Corresponding author: ricard.vilalta@cttc.es*

Compiled May 6, 2021

This paper presents an experimental evaluation of network protocols for control and management of optical networks and optical network equipment as seen in current trends. This paper presents the YANG data modelling language and its associated RESTCONF/NETCONF protocols. Later, it details multiple data models used in optical networks, such as IETF TEAS, ONF Transport API, OpenROADM and OpenConfig. It also presents multiple protocols for telemetry (i.e., YANG PUSH,gRPC and gNMI). Later, a zero touch SDN controller architecture for multiple Standard Defining Organizations (SDO) is presented, in order to support the usage of multiple protocols in a zero touch optical network. Finally, the presented protocols implementations are experimentally evaluated and compared in terms of latency and overhead. The paper explores the usage of these protocols (e.g., in research, demonstrations and open-Source optical networking projects) and provides results and recommendations for their integration in novel equipment and networks.

http://dx.doi.org/10.1364/ao.XX.XXXXXX

1. INTRODUCTION

Software Defined Network (SDN) introduced the logical decoupling of control and data planes, by providing a logically centralized architecture to dynamically control (multiple) networks. Since its definition, SDN incorporated several key concepts such as standardized management interfaces, and open interfaces. OpenFlow was the first SDN protocol, which was early extended for optical networks [1].

Optical network control and management has evolved, and nowadays it allows to understand the proper monitored data and it is able to implement automated troubleshooting, leading to what is known as zero touch optical networking. This has been reached thanks to many innovations. Multiple Standard Defining Organizations (SDO) have contributed with data modeling languages and the corresponding data models to describe a device capabilities, attributes, operations and notifications to be performed or received from a device or system. The Internet Engineering Task Force (IETF) has proposed Yet Another Next Generation (YANG) data model language [2]. This introduction has eased evolution of SDN networks thanks to the associated tools and protocols enabling complex models with complex semantics, flexible, supporting extensions and augmentations. Later, the introduction of Protocol Buffers [3] has signified a new step in data model definition. In this paper, we will present and identified how both data modeling languages have succeeded in the definition of open optical network standards, although it is hard to reach consensus.

The proposed data modeling languages have an associated transport protocol, which provides primitives to view and manipulate the data, providing a suitable encoding as defined by the data-model. Ideally, data models should be protocol independent. Each proposed transport protocol shall provide an architecture for remote configuration and control based on client / server. It should support multiple clients, provide access lists, include transactional semantics, and deploy roll-back functionalities in case of error. The Network Configuration Protocol (NETCONF) [4] was the first protocol developed specifically for YANG data models. Later, the RESTCONF protocol [5] has been presented as a feasible solution to provide the benefits of NETCONF using Representational State Transfer (REST) with Hypertext Transfer Protocol (HTTP). In the last years, a new breed of protocols for network configuration that are able to deal

with protocol buffers have been proposed, such as gRPC [6] and gNMI [7].

On the other hand, SDN-enabled applications are being developed e.g., on top of Open-Source based SDN controllers that are responsible for the logically centralized control of the optical network. These optical SDN controllers provide the necessary network simplification through abstraction and virtualization. These are provided using multiple services, such as: topology, provisioning, path computation, virtualization, and inter-domain connectivity. Several protocols have been defined for NorthBound interfaces of an optical SDN controller. The Open Networking Foundation (ONF) proposed the set of functional requirements and information model(s) for the Transport API (T-API) [8]. In parallel, the IETF working group Traffic Engineering Architecture and Signaling (TEAS) has worked in data models for Traffic Engineering (TE) topology description [9] and for Traffic Engineering connectivity provisioning [10].

As SouthBound Interfaces (SBI), optical SDN controllers have started to introduce the following data models with regard to control and management of network equipment: OpenROADM [11] and OpenConfig [12]. OpenROADM is a multi-source agreement between multiple manufacturers to make reconfigurable optical add/drop multiplexers (ROADMs) compatible across vendors. OpenConfig is an informal working group of network operators adopting SDN principles such as model-driven management. It contains both protocol and data models definition. It focuses on router and line card configuration.

Finally, network Operations require the continuous monitoring of network data provided through streaming mechanisms, which is known as telemetry. Telemetry has been using Simple Network Management Protocol (SNMP) for the last decade, both in synchronous and asynchronous monitoring. In order to introduce more significant values and lower performance overheads, several protocols are being proposed. These are YANG PUSH [13], gRPC [6], and gNMI [7].

This paper extends [14]. It provides more depth in describing the state of the art of the multiple protocols and data models involved in controlling and monitoring optical networks and elements. It also includes a description of an SDN controller architecture that is able to handle multi-SDO interfaces. This architecture has been generalized from current SDN controller implementations. Later, the concept of zero touch optical networking is introduced, including its characteristics. Finally, this paper also includes an experimental evaluation of the proposed protocols using the same information model and scenarios. Although all experimental results might not be comparable, this evaluation provides some insights on the behaviour in terms of payload length and related protocol efficiency.

The paper is organized as follows. Section 2 provides a detailed overview of current protocols for control and monitoring optical networks and equipment. Section 3 describes current attempts for SDN Controllers to implement multiple standards based interfaces, including the main characteristics of zero touch optical networking and how the used telemetry can be applied to automate optical networks. Section 4 evaluates the various presented protocols and it provides detailed results with regard to protocol bits usage and latency. Finally, Section 5 concludes.

2. CURRENT CONTROL AND MONITORING PROTOCOLS FOR OPTICAL NETWORKS AND EQUIPMENT

In this section, we provide an state of the art of current data models and protocols for: (a) YANG modelling language and protocols; (b) control and management of optical networks and equipment; and (c) telemetry mechanisms.

A. YANG modelling language and protocols

This section presents the YANG data modelling language and the NETCONF/RESTCONF protocol. Both are used for defining network control interfaces, usually available at thee NorthBound Interface (NBI) of an SDN controller. The main concepts are summarized in Table 1.

Table 1. NETCONF/RESTCONF Summary

	NETCONF	RESTCONF
Data Modelling Language	YANG	YANG
Transport	SSH, TLS, BEEP/TLS, SOAP/HTTP/TLS	HTTP
Encoding	XML	XML/JSON
Capability exchange	During Session establishment	Retrieval of Yang modules and capability URIs
Multiple datastores	YES	YES
Datastore Locking	YES	NO
Security	SSH	TLS

A.1. YANG

YANG is a data modeling language [2]. It is used to define a component configuration, state and notifications. YANG structures data into data trees within the so called datastores, by means of encapsulation of containers and lists, and to define constrained data types. It allows the refinement of models by extending and constraining existing models (by inheritance/augmentation), resulting in a hierarchy of models. A YANG model includes a header, imports, include statements, type definitions, configurations and operational data declarations as well as actions (RPC) and notifications.

Listing 1. Topology data model using YANG

```
module topology {
  namespace "urn:topology";
  prefix "topology";
  organization "CTTC";
  description "topology";
  typedef layer-protocol-name {
     type enumeration {
         enum "ETH";
         enum "OPTICAL";
     }
  }
  grouping port {
    leaf port-id {
      type string;
    }
    leaf layer-protocol-name {
      type layer-protocol-name;
    }
  }
  grouping node {
    leaf node-id {
```

```
      type string;
    }
    list port {
       key "port-id"; uses port;
    }
  }
  grouping link {
    leaf link-id {
      type string;
    }
    leaf source-node {
      type leafref {
        path "/topology/node/node-id";
      }
    }
    leaf target-node {
      type leafref {
        path "/topology/node/node-id";
      }
    }
    leaf source-port {
      type leafref {
        path "/topology/node/port/port-id";
      }
    }
    leaf target-port {
      type leafref {
        path "/topology/node/port/port-id";
      }
    }
  }
  grouping topology {
      list node {
        key "node-id"; uses node;
      }
      list link {
        key "link-id"; uses link;
      }
  }
  container topology {
    uses topology;
  }
}
```

As an example, we present Listing 1 that shows a simplified topology data model. It shows the definition of a module (topology), which consists on a container (topology) that includes a list of nodes (list node) and links (list link). Each list is ordered using its identifier (node-id, link-id, respectively). A node includes a node identifier (node-id) and a list of ports (list port). Each port has its own identifier (port-id) and a specific layer (layer-protocol-name). Each link includes a link identifier (link-id) and references (leafref) to source and target node and port identifiers.

YANG has had a significant adoption as data modeling language in Open Source projects and SDOs. There exists an ongoing significant effort to model constructs including optical devices, such as transceivers, ROADMs. This has led to literally hundreds of emerging standards across multiple SDO.

A.2. NETCONF

NETCONF is a control and management protocol that supports the configuration of devices based on their known YANG-based data models. It is based on the exchange of XML-encoded RPC messages over a secure (commonly Secure Shell, SSH) connection.

NETCONF offers operation primitives to view and manipulate data (e.g., <get-config>, <edit-config>). Data is arranged into one or multiple configuration datastores. NETCONF defines the existence of one or more datastores and allows configuration operations on them. Only the running configuration datastore is present in the base model [15].

NETCONF enabled devices include a NETCONF server, while control and management applications include a NETCONF client. Firstly, they will establish a session over a secure transport. After, both entities will send a hello message to announce their protocol capabilities, the supported data models, and the server's session identifier. Finally, when accessing configuration or state data, with NETCONF operations, subtree filter expressions can select subtrees.

One of the first evaluations of NETCONF in optical networks was presented in [16]. The authors propose a YANG model to describe a sliceable transponder to be deployed in an elastic optical network with variable rate, code, modulation formats, and monitoring capabilities. NETCONF is proposed to comply with two use cases: a) transponder configuration; and b) notification upon BER threshold exceed. The authors provide an overview of the messaging of NETCONF between client and server, which includes edit-config, subscription, and notification messages.

A.3. RESTCONF

RESTCONF is a protocol that provides an HTTP-based API to access the data, modeled by YANG [5]. RESTCONF protocol organizes the datastore using Uniform Resource Identifier (URI) that reflect data hierarchy. HTTP REST operations such as Create, Read, Update, Delete (CRUD) are applied to the defined URI.

State and Configuration data can be retrieved with the HTTP GET action. Configuration data can be modified with the POST (create), PUT (update) and DELETE methods. Data is encoded with either XML or JSON.

Current SDN controllers implement their NBI using HTTP servers. The introduction of RESTCONF to ONOS and OpenDayLight has eased the adoption of novel data models, such as the one presented in the following subsections.

B. Control and Management of Optical Networks and Equipment

This section focuses on describing the most extended YANG data models for control and management of optical network s and equipment, which have been adopted in disaggregated optical networks: IETF TEAS, ONF Transport API (T-API), OpenROADM and OpenConfig. IETF TEAS and ONF T-API use RESTCONF, while OpenROADM and OpenConfig use intensively the NETCONF protocol. Table 2 summarizes the presented standards. Complexity refers to the number of YANG lines for describing the data model. Maturity takes into consideration the number of proof-of-concept and field-trial, where the data model has been involved.

B.1. IETF TEAS

TEAS is an IETF Working Group (WG) that has proposed the YANG Data Model for Traffic Engineering (TE) Topologies [9] and for Traffic Engineering Tunnels and Interfaces [10]. The proposed data models allow network traffic engineering and they can be easily extended with optical parameters, such as in Wavelength Switched Optical Networks (WSON) [17] or FlexiGrid networks [18]. These extensions are described in Common Control and Measurement Plane (CCAMP) WG.

The authors in [19], present an analysis of different data models are compared, detailing their respective strengths and weaknesses and have shown their application areas by mapping them to components of a generic optical network SDN controller.

IETF has also proposed Abstraction and Control of TE Networks (ACTN) framework as high-level abstractions, which includes multi-technology and multi-domain transport network services. It enables legacy heterogeneous transport network control and management technologies. In order to achieve multi-domain service coordination based on abstraction/virtualization

Table 2. Standards Summary

Standards	IETF TEAS	T-API	OpenROADM	OpenConfig Data Model
Objective	NBI Transport SDN controller	NBI Transport SDN controller	Dissagregated ROADM	Router and line card configuration
Data Model	YANG	YANG	YANG	YANG
Complexity	++	+	++	+
Maturity	+	++	++	++
SDO	IETF	ONF, OIF	MSA	MSA

is provided through a hierarchy of controllers. Customer network controllers (CNC) are deployed on-demand by customers in order to handle their own virtual and abstracted resources. Multi-domain service coordinators (MDSC) are responsible to offer resources to connected CNC. MDSC are responsible for coordinating underlying physical network controllers (PNC).

In [20], the application of the ACTN architecture is demonstrated for the control of a multi-domain flexi-grid optical network. It adopts and extends the hierarchical active stateful Path Computation Element (PCE) architecture to provide per link partitioning of the optical spectrum based on variable-sized allocated frequency slots enabling network sharing and virtualization. MDSC can be mapped to parent PCE and PNC can be mapped to child PCE. Another demonstration of ACTN applicability is proposed in Cross Stratum Orchestration (CSO) as a feasible solution for NFV points of presence interconnection[21]. In the paper, the demo architecture and the interfaces/APIs were aligned with IETF ACTN.

In parallel to the IETF proposed data models, the ONF Transport API (T-API) has proven a feasible solution for multi-vendor SDN in transport networks[8]. Its usage in transport SDN controllers is spreading due to several benefits: (a) Well-known API; (b) extensible and open source software solutions; (c) tested and deployed in multiple interoperability events; (d) provides proper abstractions; (e) covers a multitude of use cases, such as multi-layer; (f) supported by a great community of vendors.

Figure 1 introduces the main ONF Transport API concepts. All interactions between an SDN controller and a T-API client (e.g., application, or orchestrator) occur within a shared context, which is defined by a set of Service Interface Points (SIP). SIPs allow a T-API client to request a connectivity services between them. An SDN controller may expose one or more abstract topologies within a shared context. These topologies may or may-not map 1-to-1 to a provider's internal topology. They are expressed in terms of nodes and links. On one side, nodes aggregate Node Edge Points (NEP). On the other side, links connect two or more nodes and they terminate on NEPs. These NEPs may be mapped to 1 or more SIP.

A T-API client might request a Connectivity Service(CS) between two or more SIPs. In response to the requested CS, the SDN controller creates one or more connections. The Connection End Points (CEP) encapsulate the related connection information regarding underlying node edge points.

B.2. ONF Transport API

The paper [22] demonstrated, for the first time, the deployment and use of ONF Transport API. Topology and connectivity services were demonstrated as a feasible solution for multi-vendor SDN in transport networks. Multiple authors have extended T-API to support multiple novel technologies, such as Space Division Multiplexing (SDM). In the paper [23], the authors have extended T-API for SDM networks and demonstrated the extensions in an experimental NFV MANO framework for the control of SDM/WDM-enabled fronthaul and packet-based transport networks.

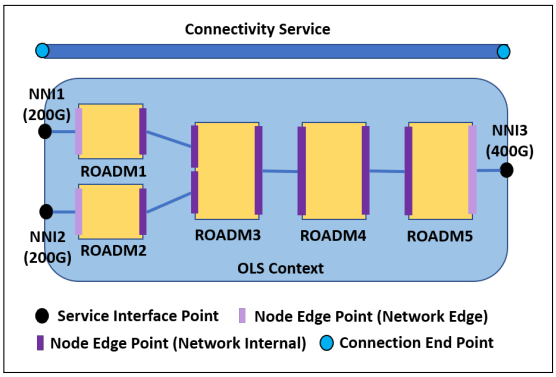

Fig. 1. Example of ONF Transport API context and main concepts

The ONOS SDN controller has demonstrated the recursive usage of Transport API on the project Open and Disaggregated Transport Network (ODTN) [24]. The aim of this project is to provide support for Ethernet (connectivity service establishment between client ports) and photonic layers (between transceiver line ports). On the other hand, OpenDayLight includes TransportPCE project, which also targets the photonic and OTN layers, including the ability to configure OTN cross-connections and interfaces. It includes a partial implementation of ONF T-API to support the export of an abstracted view of the topology to an SDN controller [25].

B.3. OpenROADM

OpenROADM is a Multi-Source Agreement (MSA) between operators and vendors that defines inter-operability specifications for ROADM switches, transponders, and pluggable optics. Specifications consist of both optical inter-operability as well as YANG data models.

OpenROADM defines vendor-neutral model for configuration and management for three network interfaces:

- common single-wave interface between transponders,

- common multi-wave interface between ROADMs,
- common YANG models for all components.

The authors in [26] show end-to-end carrier Ethernet circuits orchestration in a hierarchical control plane of ONOS SDN controllers. New device drivers were developed for ONOS in order to directly configure ROADMs using NETCONF and YANG models defined by the OpenROADM project.

[27] presented the most complete OpenROADM demonstration. It provided novel functionalities for multi-layer use cases using TransportPCE project of OpenDayLight. TransportPCE communicates with the SDN orchestrator. The SDN orchestrator provides unified control and management of the resources across three domains (i.e, optical layer, Ethernet layer, and datacenter compute nodes). Several use cases have been demonstrated, such as datacenter backups and massive virtual machine live migrations.

B.4. OpenConfig

OpenConfig is an informal working group of network operators with the common objective of introducing to the networks more programmability and dynamicity. A key driver for this objective is the adoption of SDN principles that include declarative configuration and model-driven operations.

OpenConfig provides a set of vendor-neutral YANG data models for a variety of network elements, from routers, switches to optical transport. If we focus on optical-transport, it provides a configuration and state model for:

- Terminal device: terminal optical devices within a DWDM system, including both client- and line-side parameters.
- Wavelength router: configuration and operational state data for an optical transport line system node, or ROADM (e.g., Colorless Directionless Contentionless (CDC) ROADMs, Wavelength Selective Switches (WSS), Dynamic Gain Equalizer).
- Optical amplifier: configuration and operational state data for optical amplifiers, deployed as part of a transport line system.
- Channel monitor: operational state data for an optical channel monitor (OCM) for optical transport line system elements such as wavelength routers (ROADMs) and amplifiers.
- Transport line protection: configuration and operational state data for optical line protection elements, such as Automatic Protection Switch (APS). APS provides protection using two dark fiber pairs to ensure the amplifiers can still receive a signal if one of the two fiber pairs is broken.

The paper [28] presents extensions to the OpenConfig terminal device data model that enable dynamic selection of transmission parameters of 100G/400G transmitters with coherent reception in a filter-less metro network.

The authors in [29] provide a demonstration of augmentation of the OpenConfig data model of Wavelength router to demonstrate network disaggregation, including various operations on both degrees and media channels.

The ONOS ODTN project includes OpenConfig models over NETCONF for transponder configuration. The reasons for selecting these models were that OpenConfig is a well know API that is supported already by many vendors; it provides proper abstraction model for transponder devices capabilities and information; it also defines capabilities at correct level for programmability but also abstraction from physical details; it includes capability and flexibility to support vendor specific features; and it is extensible and open source, among others.

OpenConfig and OpenROADM, although they try to provide similar solutions for modelling optical networks and elements, they are typically used for different purposes. For example, ONOS ODTN project, as previously detailed uses OpenConfig for transponder devices, but it uses OpenROADM for the OLS [30].

C. Telemetry mechanisms in Optical Networks

Streaming of monitoring data is known as telemetry and it is used in the context of network control and management, with the objective to monitor and troubleshoot network status. SNMP has been the most used protocol for this purpose [31]. Transaction Language 1 (TL1) is also an existing widely used management protocol. Several types of data are exposed and consumed in transport networks, such as network state indicators, network statistics, and critical infrastructure information. During the last decade, new protocols (i.e., YANG PUSH, gRPC, gNMI) have incorporated significant novel features that make them attractive to network administrators:

- ability to do bulk retrievals;
- support of multi-tenancy of the monitored device (e.g., through multiple NMS);
- introduction of highly efficient protocol buffers (protobuf).
- novel subscription types are defined, with the introduction of conditional telemetry, which refers to the establishment of a push connection and forward targeted telemetry data to a targeted recipient when certain criteria are met.

Table 3 provides a summary of gRPC and gNMI as protocols. YANG PUSH mechanism uses NETCONF, which has previously been detailed.

Table 3. Telemetry Summary

	gRPC	gNMI
Data Modelling Language	Protocol Buffers	YANG, (Protocol Buffers)
Transport	HTTP/2	gRPC
Encoding	byte	JSON/byte
Capability exchange	No	During Session establishment
Multiple datastores	NO	YES (Config/ State/Operational)
Datastore Locking	NO	NO
Security	TLS	TLS

C.1. YANG PUSH

YANG-PUSH is a mechanism that allows applications to subscribe to a customized stream of updates from any available YANG datastore [13]. It is provided over NETCONF protocol. It allows a client to monitor changes to a YANG datastore with notifications instead of with NETCONF GET requests. This provides lower resource consumption and ease of system integration.

Multiple datastores can be selected for subscription (e.g., running, candidate, operational). There are 2 kinds of datastore subscriptions:

- Periodic subscriptions: The data is sent to the client every specified time interval, even if the data is not modified. Periodic subscriptions are similar to periodic NETCONF GET requests, but they differ in the fact that the client does not send any request.
- On-change subscriptions: The data is sent to the client only when the data changes. This subscription method is much more efficient than the periodic requests or subscription approaches.

The work presented in [32] demonstrated for the first time the implementation of YANG push notifications on an open terminal based on OpenConfig models. Examples for both periodic and on-change subscription are provided.

C.2. gRPC

Google Remote Procedure Calls (gRPC) [6] is a protocol based on HTTP/2 as a transport protocol and it uses protocol buffers encodings for transported messages. As it is based on HTTP/2 transport protocol and uses byte-oriented encoding, it introduces low latency. gRPC has been used in highly scalable and distributed systems.

Protocol Buffers are a language-neutral, platform-neutral extensible mechanism for serializing structured data [33]. They allow to model the data structures in a similar manner as in YANG. Its encoding in byte oriented messages increases the efficiency compared to XML/JSON encodings. An Open-Source framework is provided to handle protocol buffers in multiple languages, such as python, Go, C++, Java.

Listing 2 provides an example of the same topology information model previously presented using YANG. In this example, a topology service is described and it offers a GetTopology RPC that provides a topology structure. Each structure is described in message keyword. Topology message provides a list of node messages (a list is declared using repeated keyword) and a list of link messages. A node message includes an identifier and a list of port messages. Port messages include a port identifier and a layer protocol name. Link messages include a link identifier and references to source and target nodes and ports.

The authors in [34] provide the first experimental demonstration of gRPC-based SDN control and telemetry architecture. They apply gRPC to control and monitor spectral/spatial superchannels with SDM/WDM transceivers monitored the BER to detect soft-failures over an 11-km 6-mode 19-core fiber.

Listing 2. Topology service described using protocol buffers

```
//Example of topology
syntax = "proto3";
package topology;

import "google/protobuf/empty.proto";

service TopologyService {
  rpc GetTopology (google.protobuf.Empty) returns (Topology) {}
}


message Link {
  string link_id = 1;
  string source_node = 2;
  string target_node = 3;
  string source_port = 4;
  string target_port = 5;
}

message Node {
  string node_id = 1;
  repeated Port port = 2;
}

message Port {
  string port_id = 1;
  enum LayerProtocolName {
    ETH = 0;
    OPTICAL = 1;
  }
  LayerProtocolName layerProtocolName = 2;
}

message Topology {
  repeated Node node = 1;
  repeated Link link = 2;
}
```

C.3. gNMI

gRPC Network Management Interface (gNMI) [7] is a protocol for configuration manipulation and state retrieval. It is built on top of gRPC and it is described using protobuf and it can use binary or JSON encoding for payload. This allows the usage of YANG data models, allowing the integration of all efforts for defining them in SDOs.

A gNMI target is the device which acts as the owner of the data that is being manipulated or reported on. Typically this will be a network device. A gNMI client is the device using the protocol to query/modify data on the target. Typically this will be a network management system.

gNMI telemetry makes use of the SubscribeRequest message. It has been demonstrated to assess and retrieve transmission performance and rapidly determine the most suitable operational mode in [35].

The authors in [36] have demonstrated optical performance monitoring data across the optical terminal and OADM devices. These data is sent to an analysis collection platform. OpenConfig usage with gNMI has been demonstrated by the authors in [37]. In this paper, real-time monitoring of optical transponders using gNMI is demonstrated using a Transport SDN controller.

3. MULTI-SDO SDN CONTROLLER ARCHITECTURE AND ZERO TOUCH OPTICAL NETWORKING

This sections presents a generalized architecture that is able to handle multiple data models, as well as it later presents how to use the presented architecture to provide zero touch optical networking.

A comprehensive picture of Transport SDN is provided by [38], where it is provided an analysis of the multiple SDN controllers. Current SDN controllers, such as OpenDayLight or ONOS, allow the interaction with other SDN controllers and network equipment using multiple defined data models from multiple SDOs. We refer to this kind of SDN controller as multi-SDO SDN controller.

In this section we generalize the common functionalities that provide current SDN Controllers to overcome with multiple

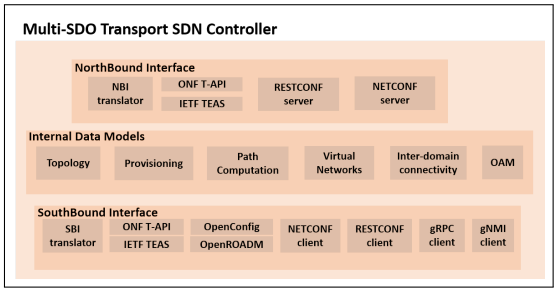

Fig. 2. Multi-SDO optical SDN controller architecture.

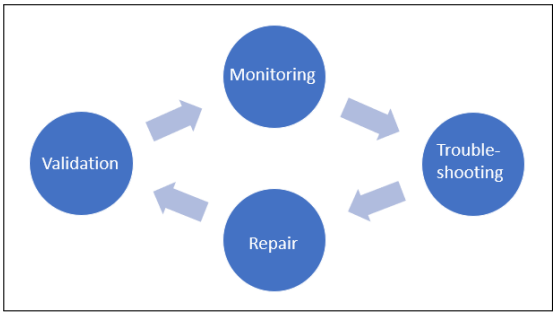

Fig. 3. Optical Zero Touch Networking state diagram.

data models and protocols. The multi-SDO SDN controller has its internal data models. They define the main purposes and activities of the SDN controller. The internal data model can be based on a well known data models (in Figure 2 ONF Transport API has been adopted as internal data model). Typical SDN controller modules and thus data models, might include topology, provisioning, path computation, virtual networks, inter-domain connectivity and operations, administration and management (OAM).

For the multi-SDO northbound interface, the NBI translator is needed, as well as the multiple servers that offer multiple protocols, such as NETCONF and RESTCONF. These servers are responsible for offering the multiple data models that can be obtained through NBI translator from the internal data models. Figure 2 shows how ONF T-API and IETF TEAS are offered as NBI using translation from internal data models.

Regarding southbound API, the mechanism is similar. Firstly, multiple clients are provided for the proposed protocols (e.g., ONF T-API, IETF TEAS, OpenROADM, and OpenConfig). Also, an SBI translator is needed to translate the required internal data model actions, towards underlying networks and equipment.

Zero touch optical networking (ZTON) is related to two main topics: a) sustain fast network growth automatically without human intervention; and b) minimize human errors (and interventions) by minimizing associated outages [6]. Thus, ZTON requires the proposed control and telemetry protocols for optical network monitoring, in order to receive proper network notifications and to interact with new configuration for network elements.

Zero touch optical networking includes the following characteristics:

- The only required operator step is the instantiating of intent. Then all network operations are automated.
- Network-wide intents are decomposed into declarative and vendor-neutral configuration for individual network elements.
- If unintended behaviour is detected, any network changes are automatically halted and rolled-back.
- Operation violating network policies shall not be allowed by the infrastructure.

Figure 3 shows the suggested state diagram [6] for providing zero touch optical networking. It consists on a closed loop that is fed with the proper monitoring data. This data is obtained through the multiple mechanisms detailed in previous telemetry section. The analysis of the data leads to error detection, which triggers automated troubleshooting. This will lead to a repair action, which requires automated control of the optical network and the declaration of novel intents for repairing it. Finally, the repair will be verified.

4. EXPERIMENTAL EVALUATION

In this section, we propose a methodological approach to evaluate the previously presented protocols that have been demonstrated to be applied to optical networks: (a) NETCONF; (b) RESTCONF; (c) gRPC; and (d) gNMI. To perform this evaluation, two topologies have been defined in order to evaluate the different protocols: a) National Science Foundation Network (NSFNET) that includes 14 nodes and 21 links; and b) European Optical Network (EON) that includes 19 nodes and 38 links. Both topologies have been defined using a JSON file that can be parsed by the multiple servers that we have created for evaluation.

The previously presented topology data models, Listings 1 and 2, have been used in order to describe NSFNET and EON scenarios. The defined data models include the same amount of information. Listing 1 has been validated using PYANG [39], while Listing 2 has been validated using Protobuf compiler [33]. All measurements have been obtained using a Ubuntu Linux Server with i7 10th Gen. CPU and 4GB RAM. Each of the experiments has been repeated 10 times and mean value is provided.

The purpose of this experiments is to evaluate numerically the proposed protocols in terms of bit usage and latency. Bit usage refers to the total amount of bits interchanged between a client and a server to provide a complete request/response. Latency refers to the required round-trip time since the issue of a request action from the client and the necessary server time to answer to that action. The authors believe that for simple use cases, protocol overhead and latency might be enough for comparison, but the results are presented individually for each protocol, as each protocol provides unique features for specific use cases.

A. NETCONF

In order to evaluate NETCONF performance, the NETCONF server includes PYANGBIND [40] generated code, in order to properly describe the selected topology using the defined YANG data model (Listing 1). For NETCONF protocol, we have based our server and client implementations on opensource library NETCONF [15]. NETCONF protocol includes SSH secure connection between client and server, thus results might not be comparable with other protocols that include different security layers, such as TLS.

Figure 4 shows the results of bit usage in NETCONF get-config operation for the described topology. This operation requires 140000 bits for NSFNET topology, while it requires up to 200640 bits for EON topology. The EON topology requires more than 43% in comparison with NSFNET.

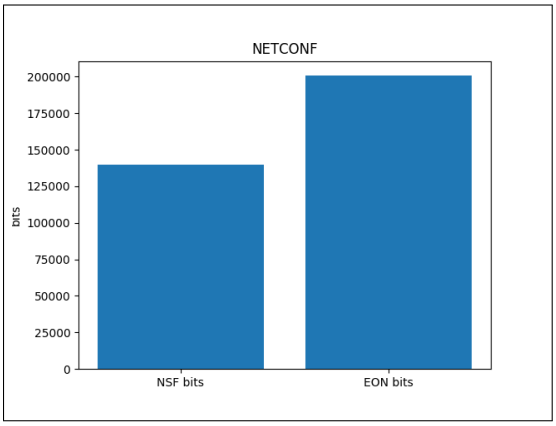

Fig. 4. Get-config NETCONF Topology evaluation. Measured bits

Figure 5 provides the measured latency, results for NSFNET and EON topologies are very similar, 6111 ms and 6172 ms, respectively, with an insignificant difference of 0.9%. Latency values are really high, due to the selected NETCONF library and implementation. It can be observed that data is exchanged 5.10 seconds after SSH session has been established.

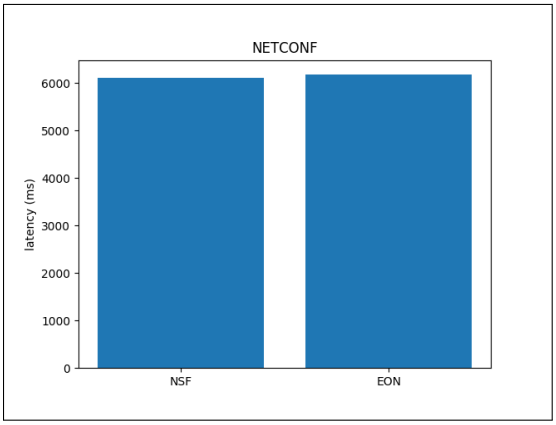

Fig. 5. Get-config NETCONF Topology evaluation. Measured latency (ms)

B. RESTCONF

In order to evaluate RESTCONF, we have obtained the RESTCONF-based Swagger API definition of the topology YANG using the YANG2SWAGGER tool from ONF [41]. Once the swagger API has been obtained, using Swagger Code generator [42], we have obtained the necessary RESTCONF server, that provides NFSNET or EON topologies, depending on configuration.

Figure 6 shows the results of bit usage in RESTCONF HTTP GET operation for each proposed topology. This operation requires 112000 bits for NSFNET topology, while it requires up to 196000 bits for EON topology. The EON topology requires more than 75% in comparison with NSFNET. This difference can be explained by the difference also in the bit usage between NSFNET and EON topologies. In Figure 6, we can also observe the results for using RESTCONF with a TLS security layer. HTTPS GET NSFNET topology requires 136000 bits, increasing the necessary bits due to TLS by 21%. For EON topology, it requires 216000 bits, representing an increase of 10%.

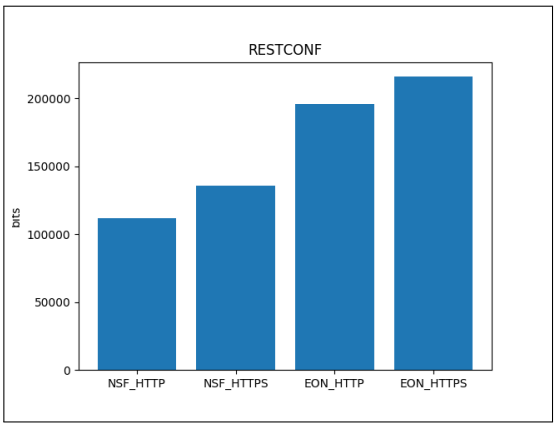

Fig. 6. HTTP/HTTPS GET operation result evaluation. Measured bits

Figure 7 provides measured latency results. For NSFNET and EON topologies results are very similar, 2.883 ms and 5.109 ms, respectively, with a significant difference of 73%. If we compare RESTCONF HTTPS latency results for NSFNET and EON are similar, 6.855 ms and 7.517 ms, respectively. The increase of latency due to TLS is of 137% and 47%, respectively.

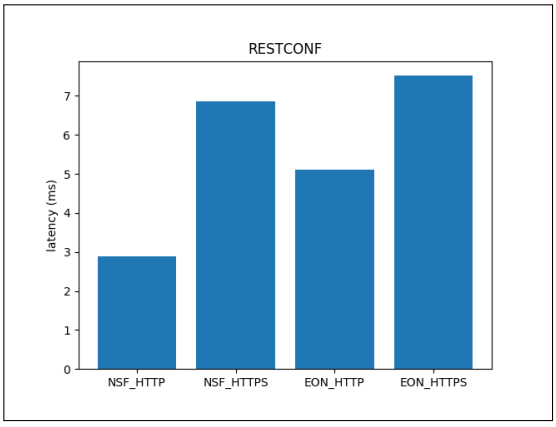

Fig. 7. HTTP/HTTPS GET operation result evaluation. Measured latency (ms)

C. gRPC

With the purpose of evaluating gRPC protocol, we will use the Listing 2 that includes the protocol buffer definition of the topology service. The described service is equivalent to the previously used YANG-based one. To compile the used protocol buffer and provide client and server stubs, the tools in [43] have been used. The same NSFNET and EON topologies have been used for the evaluation of the protocol.

Figure 8 shows the results of bit usage in gRPC RPC Get-Topology operation for each proposed topology. This operation

requires 21600 bits for NSFNET topology, while it requires up to 28000 bits for EON topology. It can be observed that protocol buffer encoding reduces 81% and 86% bit usage in NSFNET and EON topologies, respectively. In Figure 8, we can also observe the results for using gRPC with a TLS security layer. gRPC with TLS NSFNET GetTopology requires 44000 bits, increasing the necessary bits due to TLS by 104%. For EON topology, it requires 50400 bits, representing an increase of 180% , in comparison with EON topology with gRPC without TLS.

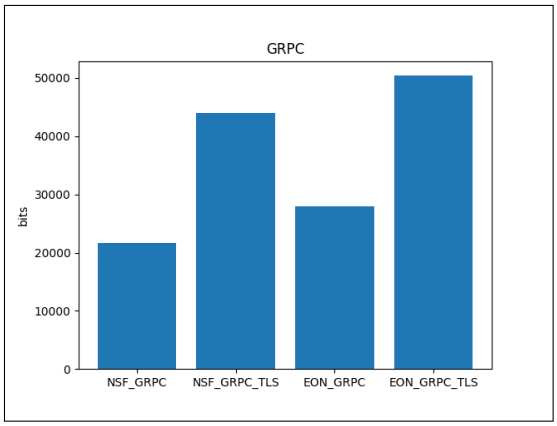

Fig. 8. gRPC RPC operation result evaluation to get topology. Measured bits

Figure 9 described the measured latency results. For NSFNET and EON topologies results are very similar, 1.159 ms and 1.430 ms, respectively. In comparison with RESTCONF protocol there is a significant difference of 60% and 72%, respectively. Results for gRPC with TLS in NSFNET and EON are similar, 26.47 ms and 30.75 ms, respectively. The increase of latency due to TLS is very significant, as it increases an order of magnitude.

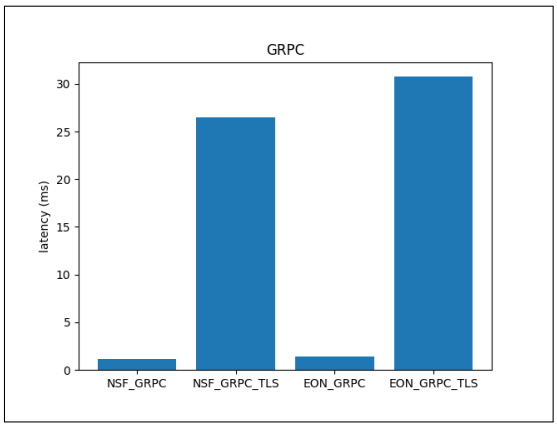

Fig. 9. gRPC RPC operation result evaluation to get topology. Measured latency (ms)

D. gNMI

Using YANG Go Tools (YGOT) [44], we generate the necessary Go language bindings for the previously presented YANG topology Listing 1. The proposed library uses JSON encoding. Once obtained, gNMI target (server) is executed using the NSFNET and EON topologies. GNMI Get client is used to retrieve the topological information. Results might be different in gNMI from obtained in gRPC, as different libraries and programming languages have been used. We leave for further exploration to evaluate gNMI using as basis gRPC python libraries.

Figure 10 shows the results of bit usage in gNMI GET operation for each proposed topology. This operation requires 48800 bits for NSFNET topology, while it requires up to 78400 bits for EON topology. It can be observed that although JSON is used for exchange, the introduction of gNMI protobuf reduces both 60% bit usage in comparison with RESTCONF results in NSFNET and EON topologies. In Figure 10, we can also observe the results for using gNMI with a TLS security layer. gNMI GET requires 78400 bits, increasing the necessary bits due to TLS by 60%. For EON topology, it requires 106400 bits, representing an increase of 36%.

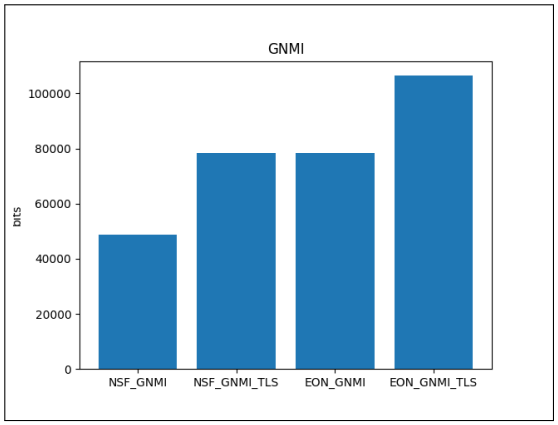

Fig. 10. gNMI GET RPC to retrieve topology. Measured bits

In Figure 11 results are shown for latency in NSFNET and EON topologies are very similar, 2.795 ms and 3.277 ms, respectively. In comparison with RESTCONF protocol there is a decrease of 3% and 36%, respectively. Regarding gNMI with TLS, latency results for NSFNET and EON are similar, 8.132 ms and 8.550 ms, respectively. The increase of latency due to TLS is of 190% and 161%, respectively, in comparison without TLS.

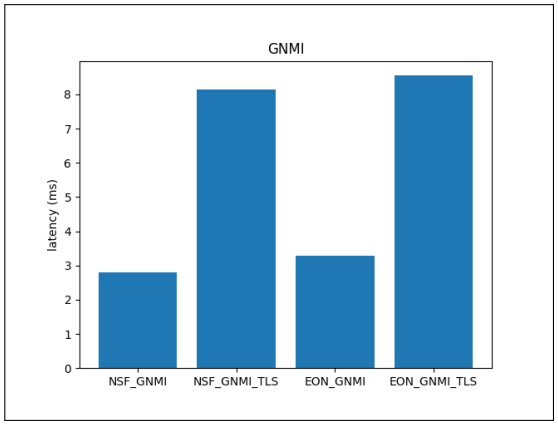

Fig. 11. gNMI GET RPC to retrieve topology. Measured latency (ms)

E. Summary

Table 1 and Table 3 present an overview of the presented and evaluated protocols. Firstly, they analyse the supported data

modelling languages. NETCONF and RESTCONF protocols support only YANG, while gRPC only supports protocol buffers. gNMI is the only protocol able to handle both data models. The tables also detail the transport protocol stack, which has been detailed in the respective protocol subsections. Thirdly, the message encoding is referenced. NETCONF supports XML encoding, while RESTCONF uses both JSON and XML. gRPC uses byte encoding and gNMI supports both JSON and byte encoding. Later, exchange of capabilities (supported data models) is detailed. Another important feature is support for multiple datastores (which allows to distinguish between config, state or operational data). Datastore locking possibilities is a key feature to allow multi-tenant support for devices, as well as multi-access configuration. Unfortunately, this feature is only supported (still) by NETCONF. Finally, the supported security protocols are described. NETCONF uses always SSH, while the rest support TLS as optional.

Table 2 shows a summary table regarding the presented data models for control and management of optical networks and equipment. ONF Transport API and IETF TEAS compete as feasible solutions for NBI of a transport SDN controller. OpenROADM focuses on control and management of disaggregated ROADMs (although in latest versions, it provides control for optical network control and management) and OpenConfig focuses on router and line card configuration. All models are described using YANG, thus we can deduce that it is widely accepted for industry. Complexity of the model can be evaluated though the learning curve of the model, as well as the documentation needed for adopting the selected data model, including the length of the data model itself. Transport API and OpenConfig provide simpler (and shorter) data models. Maturity can be evaluated by the stage of implementation of the data model, starting from demonstration, up to inter-operability tests and production deployments. Finally, we detail the type of agreement, if it is produced inside an SDO or a Multi-source agreement (MSA).

5. CONCLUSION

This paper has justified the need for data models and protocols in order to control and monitor optical networks and optical network elements.

The authors have reviewed the current state of the art with significant contributions to each of the proposed solutions. The introduced protocols (NETCONF, RESTCONF, gRPC and gNMI) have been evaluated with the same data models and under the same scenarios in order to provide numerical insights for each protocol in terms of payload length and latency. This information is required in order to better propose optical networks and equipment that shall be adapted to the necessary use cases and scenarios.

The authors have also introduced the concept of multi-SDO SDN controller, as well as presented the main characteristics of zero touch optical networking.

For further study, further results can be obtained by having some experiments on real optical networks and devices. These results will help our understanding of the proposed protocols performance when a real optical data plane is being used and how the data plane can affect their performance.

FUNDING

Work partially funded by the EC through the 5GPPP INSPIRE-5GPlus (871808), by the Spanish AURORAS (RTI2018-099178-B-I00) project and the Research and Development of Innovative Optical Network Technology for a Novel Social Infrastructure (JPMI00316), the Ministry of Internal Affairs and Communications, Japan.

REFERENCES

1. M. Channegowda, R. Nejabati, M. R. Fard, S. Peng, N. Amaya, G. Zervas, D. Simeonidou, R. Vilalta, R. Casellas, R. Martinez *et al.*, "Experimental demonstration of an openflow based software-defined optical network employing packet, fixed and flexible dwdm grid technologies on an international multi-domain testbed," Opt. express **21**, 5487–5498 (2013).
2. M. Bjorklund, "The yang 1.1 data modeling language," Tech. rep., RFC 7950, DOI 10.17487/RFC7950, August 2016,< https://www. rfc-editor. org . . . (2016).
3. G. Kaur and M. M. Fuad, "An evaluation of protocol buffer," in *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon),* (IEEE, 2010), pp. 459–462.
4. R. Enns, M. Bjorklund, and J. Schoenwaelder, "Netconf configuration protocol," Tech. rep., RFC 4741, December (2006).
5. A. Bierman, M. Bjorklund, and K. Watsen, "Restconf protocol-rfc 8040," RFC Ed. (2017).
6. E. Breverman, N. El-Sakkary, T. Hofmeister, S. Ngai, A. Shaikh, and V. Vusirikala, "Optical zero touch networking—a large operator perspective," in *2019 Optical Fiber Communications Conference and Exhibition (OFC),* (IEEE, 2019), pp. 1–3.
7. B. Claise, J. Clarke, and J. Lindblad, *Network Programmability with YANG: The Structure of Network Automation with YANG, NETCONF, RESTCONF, and GNMI* (Addison-Wesley Professional, 2019).
8. C. Janz, L. Ong, K. Sethuraman, and V. Shukla, "Emerging transport sdn architecture and use cases," IEEE Commun. Mag. **54**, 116–121 (2016).
9. X. Liu, I. Bryskin, V. Beeram, T. Saad, H. Shah, and O. Gonzalez De Dios, "Yang data model for traffic engineering (te) topologies," RFC 8795 (2020).
10. T. Saas *et al.*, "A yang data model for traffic engineering tunnels, label switched paths and interfaces," draft-ietf-teas-yangte, work progress (2021).
11. E. Riccardi, P. Gunning, O. G. de Dios, M. Quagliotti, V. Lopez, and A. Lord, "An operator view on the introduction of white boxes into optical networks," J. Light. Technol. **36**, 3062–3072 (2018).
12. A. Shaikh, T. Hofmeister, V. Dangui, and V. Vusirikala, "Vendor-neutral network representations for transport sdn," in *2016 Optical Fiber Communications Conference and Exhibition (OFC),* (IEEE, 2016), pp. 1–3.
13. A. Clemm and E. Voit, "Subscription to yang notifications for datastore updates," Tech. rep., IETF RFC 8641, DOI 10.17487/RFC8641, September 2019 (2019).
14. R. Vilalta, N. Yoshikane, R. Casellas, R. Martínez, T. Tsuritani, I. Morita, and R. Muñoz, "Controlling and monitoring optical network equipment in optical sdn networks," in *2020 European Conference on Optical Communication (ECOC),* (IEEE, 2020), pp. 1–3.
15. "Netconf: Netconf server and client library for python," https://github.com/choppsv1/netconf/. Accessed: 2021-02-28.
16. M. Dallaglio, N. Sambo, F. Cugini, and P. Castoldi, "Control and management of transponders with netconf and yang," IEEE/OSA J. Opt. Commun. Netw. **9**, B43–B52 (2017).
17. H. Zeng, Y. Lee, A. Guo, V. López, and D. King, "A yang data model for wson (wavelength switched optical networks)," draft-ietf-ccamp-wson-yang-28, work progress (2020).
18. J. Lopez de Vergara, D. Perdices, D. King, Y. Lee, and H. Zheng, "A yang data model for flexi-grid optical networks)," draft-ietf-ccamp-flexigrid-yang-09, work progress (2021).
19. T. Szyrkowiec, A. Autenrieth, and W. Kellerer, "Optical network models and their application to software-defined network management," Int. J. Opt. **2017** (2017).
20. R. Casellas, R. Vilalta, R. Martínez, R. Muñoz, H. Zheng, and Y. Lee, "Experimental validation of the actn architecture for flexi-grid optical networks using active stateful hierarchical pces," in *2017 19th Interna-*

tional Conference on Transparent Optical Networks (ICTON), (IEEE, 2017), pp. 1–4.
21. R. Vilalta, V. López, Y. Lee, H. Zheng, Y. Lin, R. Casellas, O. González-de Dios, R. Martínez, and R. Muñoz, "Towards ip & transport network transformation using standardized transport northbound interfaces," in *2018 Optical Fiber Communications Conference and Exposition (OFC),* (IEEE, 2018), pp. 1–3.
22. A. Mayoral, R. Vilalta, R. Muñoz, R. Casellas, R. Martinez, M. S. Moreolo, J. M. Fabrega, S. Yan, A. Aguado, E. Hugues-Salas *et al.*, "First experimental demonstration of a distributed cloud and heterogeneous network orchestration with a common transport api for e2e services with qos," in *Optical Fiber Communication Conference,* (Optical Society of America, 2016), pp. Th1A–2.
23. R. Vilalta, R. Muñoz, G. Landi, L. Rodriguez, M. Capitani, R. Casellas, and R. Martínez, "Experimental demonstration of the bluespace's nfv mano framework for the control of sdm/wdm-enabled fronthaul and packet-based transport networks by extending the tapi," in *2018 European Conference on Optical Communication (ECOC),* (IEEE, 2018), pp. 1–3.
24. A. Campanella, B. Yan, R. Casellas, A. Giorgetti, V. Lopez, Y. Zhao, and A. Mayoral, "Reliable optical networks with odtn: Resiliency and fail-over in data and control planes," J. Light. Technol. **38**, 2755–2764 (2020).
25. M. Birk, O. Renais, G. Lambert, C. Betoule, G. Thouenon, A. Triki, D. Bhardwaj, S. Vachhani, N. Padi, and S. Tse, "The openroadm initiative," IEEE/OSA J. Opt. Commun. Netw. **12**, C58–C67 (2020).
26. R. Morro, F. Lucrezia, P. Gomes, R. Casellas, A. Giorgetti, L. Velasco, E. Riccardi, A. C. Piat, A. Percelsi, J. Pedro *et al.*, "Automated end to end carrier ethernet provisioning over a disaggregated wdm metro network with a hierarchical sdn control and monitoring platform," in *2018 European Conference on Optical Communication (ECOC),* (IEEE, 2018), pp. 1–3.
27. B. Mirkhanzadeh, S. Vachhani, B. G. Bathula, G. Thouenon, C. Betoule, A. Triki, M. Birk, O. Renais, T. Zhang, M. Razo *et al.*, "Demonstration of joint operation across openroadm metro network, openflow packet domain, and openstack compute domain," in *Optical Fiber Communication Conference,* (Optical Society of America, 2020), pp. W3C–3.
28. F. Paolucci, A. Sgambelluri, R. Emmerich, A. Giorgetti, P. Castoldi, C. Schubert, J. K. Fischer, and F. Cugini, "Openconfig control of 100g/400g filterless metro networks with configurable modulation format and fec," in *Optical Fiber Communication Conference,* (Optical Society of America, 2019), pp. Tu3H–4.
29. L. Dou *et al.*, "Demonstration of open and disaggregated roadm networks based on augmented openconfig data model and node controller," in *2020 Optical Fiber Communications Conference and Exhibition (OFC),* (IEEE, 2020), pp. 1–3.
30. R. Casellas, R. Vilalta, R. Martínez, and R. Muñoz, "Sdn control of disaggregated optical networks with openconfig and openroadm," in *International IFIP Conference on Optical Network Design and Modeling,* (Springer, 2019), pp. 452–464.
31. D. Harrington, R. Presuhn, and B. Wijnen, "Rfc3411: An architecture for describing simple network management protocol (snmp) management frameworks," (2002).
32. A. Mayoral, V. Lopez, J. P. Fernández-Palacios, Y. Yufeng, Z. Lifen, H. Wenjun, Z. Mingfeng, and Y. Changlong, "First demonstration of yang push notifications in open terminals," in *2020 International Conference on Optical Network Design and Modeling (ONDM),* (IEEE, 2020), pp. 1–3.
33. "Protocol buffers," https://github.com/protocolbuffers/protobuf. Accessed: 2021-02-28.
34. R. Vilalta, N. Yoshikane, R. Casellas, R. Martínez, S. Beppu, D. Soma, S. Sumita, T. Tsuritani, I. Morita, and R. Muñoz, "Grpc-based sdn control and telemetry for soft-failure detection of spectral/spacial superchannels," in *45th European Conference on Optical Communication (ECOC 2019),* (IET, 2019), pp. 1–4.
35. A. Sgambelluri, F. Paolucci, and F. Cugini, "Telementry-driven validation of operational modes in openconfig disaggregated networks," in *ECOC 2019,* (2019).
36. A. Sadasivarao, S. Jain, S. Syed, K. Pithewan, P. Kantak, B. Lu, and L. Paraschis, "High performance streaming telemetry in optical transport networks," in *Optical Fiber Communication Conference,* (Optical Society of America, 2018), pp. Tu3D–3.
37. R. Vilalta, C. Manso, N. Yoshikane, R. Muñoz, R. Casellas, R. Martínez, T. Tsuritani, and I. Morita, "Telemetry-enabled cloud-native transport sdn controller for real-time monitoring of optical transponders using gnmi," in *2020 European Conference on Optical Communication (ECOC),* (IEEE, 2020), pp. 1–3.
38. R. Alvizu, G. Maier, N. Kukreja, A. Pattavina, R. Morro, A. Capello, and C. Cavazzoni, "Comprehensive survey on t-sdn: Software-defined networking for transport networks," IEEE Commun. Surv. & Tutorials **19**, 2232–2283 (2017).
39. "Pyang: an extensible yang validator and converter," https://github.com/mbj4668/pyang. Accessed: 2021-02-28.
40. "Pyangbind," https://github.com/robshakir/pyangbind. Accessed: 2021-02-28.
41. "Yang2swagger generator," https://github.com/bartoszm/yang2swagger. Accessed: 2021-02-28.
42. "Swagger codegen," https://github.com/swagger-api/swagger-codegen. Accessed: 2021-02-28.
43. "grpc quick start," https://grpc.io/docs/languages/python/quickstart/. Accessed: 2021-02-28.
44. "gnxi tools," https://github.com/google/gnxi. Accessed: 2021-02-28.