# A Non-Proprietary Network Operations Platform for OpenROADM Environment

**Nathan Ellsworth[1], Behzad Mirkhanzadeh[1], Tianliang Zhang[1], Shweta Vachhani[2], Balagangadhar Bathula[2], Gilles Thouenon[3], Christophe Betoule[3], Olivier Renais[3], Miguel Razo[1], and Andrea Fumagalli[1]**

[1] *Open Networking Advanced Research (OpNeAR) Lab, UT Dallas, TX, USA.* [2] *AT&T Labs, 200 Laurel Avenue South, Middletown, NJ, USA.* [3] *Orange Labs, 2 Avenue Pierre Marzin, Lannion, France*

*Email: nathan.ellsworth@utdallas.edu*

**Abstract:** Key functionalities of NOP (Network Operations Platform) are demonstrated with the latest multi-vendor OpenROADM equipment. Using open source packages, the NOP inter-operates with TransportPCE and other controllers, bringing together information about topology, events, and metrics.

**OCIS codes:** (060.4250) Networks, (060.2310) Fiber optics. 

## 1. Overview

At the heart of nearly any modern enterprise is a room filled with large screens full of important information being watched by specialists who have the mandate to keep the computer and network assets performing at their designed specifications. This place is variously referred to as the Network Operations Center (NOC), Command Center (CC), or other such terms. The operators in the NOC use the information at their fingertips to prevent potential outages or to restore services as quickly as possible after an outage.

NOC data feeds are traditionally classified as FCAPS for *Fault*, *Configuration*, *Accounting*, *Performance*, and *Security* [1]. We have chosen a simpler but similar grouping on which to focus: *Topology*, *Events*, and *Metrics*. Events (including both status and alarms) would fall under Faults, metrics under Performance, and topology under Configuration. This is a semantic breakdown that helps organize our efforts, but arguments could be made for different groupings.

In the literature there has been some discussion of FCAPS and monitoring related to disaggregated optical networking [2]. However, we have not seen much work live-demonstrated or published in this area. We plan to demonstrate our addition of a *Network Operations Platform* or *NOP* to the PROnet multi-vendor OpenROADM environment shown in Fig. 2. The environment includes OpenROADM equipment — controlled through TransportPCE — that operates at multiple data rates and is connected to OpenFlow-enabled switches and compute nodes controlled through OpenStack and Kubernetes. The TransportPCE, OpenStack, Kubernetes, and other network components are all controlled by the PROnet orchestrator software module [3].

The NOP provides functions similar to what would be present in a traditional NOC. It graphically presents to the user the topology relationships among the network and compute components, a real-time view of important events such as alarms or service updates, and key performance indicators (KPIs, or metrics).

## 2. Design

### 2.1. Architecture

Figure 1 shows a basic block-level diagram of two data center PROnet environment (left and bottom) overlaid with the NOP module (top right). Within the NOP module are three pipelines that each have instrumentation points to bring in information. These are color-coded as blue for events, green for metrics, and yellow for topology.

### 2.2. Pipelines

**Topology.** In a NOC, it is important for the operators to have a clear picture of how various elements are related in order to reduce service outages. This view should be dynamic, based on actual physical topology as discovered from the devices themselves. Using a REST API call, we can gather the topology TransportPCE has discovered via NETCONF from the optical equipment. OpenStack and Kubernetes can also give us topology information about configured hosts, guests, and services. Each discovered device or service is a node and each discovered connection is an edge and these node-edge pairs are stored in an open-source graph database based on neo4j [4]. The graph query language GraphQL [5] is used to present this topology graphically to the user. TransportPCE itself has a UI component which may be used in addition to neo4j.

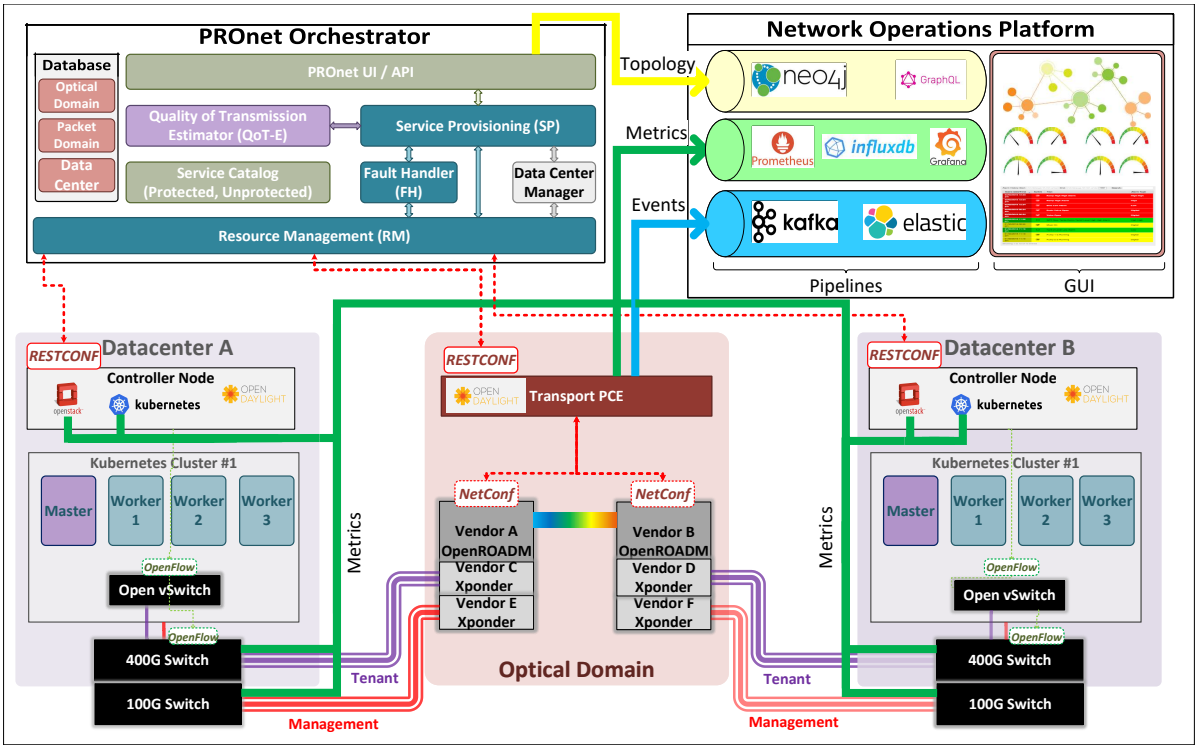


Fig. 1. NOP key open source modules and related interfaces toward TranportPCE, ODL, OpenStack, Kubernetes, and PROnet Orchestrator. Events flow is blue, metrics are green, topology is yellow.

**Events.** Most of the pieces of our environment produce detailed log data that are tedious to parse by hand. We will feed all useful logs into the standard ELK stack (made of Logstash, Elastisearch, Kibana) [6]. A critical new addition our work brings is exposure of internal status information from TransportPCE. A new version of TransportPCE publishes this information via a Kafka message bus [7], allowing the NOP to subscribe to real-time asynchronous updates on the status of lightpath service requests. These Kafka-sourced messages are also stored in the ELK repository. The Logstash repository can keep this event data in a compressed and indexed format for as long as needed. This will be useful for follow-on analysis work looking for patterns and trends in the past that might predict behavior in the future.

**Metrics/KPIs.** While the event pipeline focuses on textual information, the metric pipeline processes numerical metrics that are also available from many points in the system. A number of standard software products, including OpenStack, Kubernetes, and Ubuntu, are enabled to be monitored out-of-the-box with Prometheus [8], tracking important metrics such as CPU, memory, disk, and network utilization. Prometheus can also bring in Network switch and router metrics via SNMP or streaming telemetry mechanisms. This data is stored in a time-series InfluxDB [9] database, which is purposely designed to achieve long-term storage and retrieval of time-stamped data at sub-second resolution. This function will enable use-cases such as detection of bursty traffic on high-speed switch ports. We also capture different optical related values, such as lightpath frequency, signal bandwidth and fiber loss. The Grafana [10] graphical reporting engine is used to present this data to the user in either pre-configured dashboards or in an ad-hoc fashion.

### *2.3. Implementation*

The implementation of the NOP module neatly overlays on top of the existing OpenROADM physical network gear hosted at UT Dallas. Fig. 2 shows the various components from the block diagram in Fig. 1. The NOP runs as a collection of containers in the management server area. Note, however, that the NOP does not have to be in immediate proximity to the devices under monitoring. In fact, the GUI presentation layer, being entirely web-based, can be shown from nearly anywhere with network access.

## 3. Innovation

The innovation of this work is as follows:

1. Built exclusively using non-proprietary open source software packages, as discussed in the paper. This is in

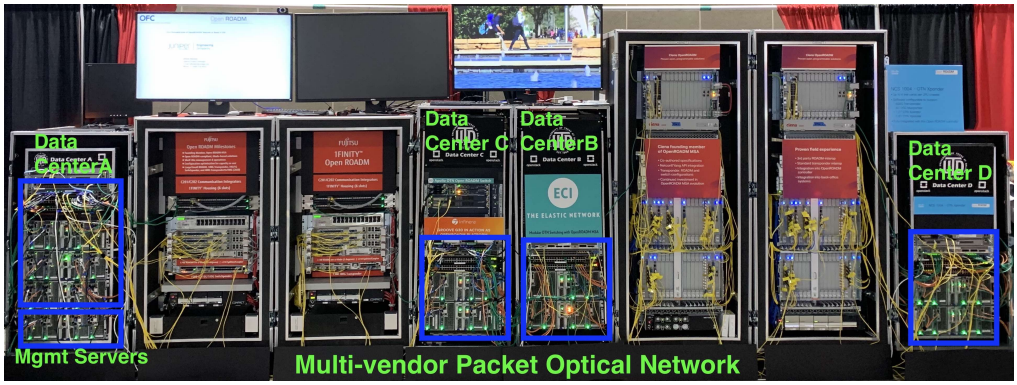


Fig. 2. OpenROADM environment (as demonstrated at OFC 2020 [3]) showing location of management servers (bottom left) in relation to Datacenters A, B, C, and D with packet optical multi-vendor OpenROADM network in between.

line with our ongoing work with the OpenROADM MSA and other standards.

2. Utilizes TransportPCE status and device alarm data, particularly lightpath establishment messages, which has not been done before. In prior iterations of the UT Dallas OpenROADM demonstration, this lightpath status had to be be gleaned by synchronous polling.

3. Brings together additional key information from other parts of the environment to provide a more-complete view of the environment, appropriate for a NOP.

4. Makes use of the latest OpenROADM equipment from a number of equipment vendors (Fig. 2 shows an *example* of the equipment that is going to be connected to the NOP demo at OFC'21).

## 4. OFC Relevance

Open and disaggregated optical transport network solutions continue to gain traction. The NOP demonstration addresses the important aspect of providing FCAPS capabilities to OpenROADM compliant equipment [2]. This objective is achieved through a number of open-source packages that are configured to operate with the open-source TransportPCE controller. Additional functionalities are also provided in the NOP demonstration to illustrate scalability of the proposed open-source platform to include other domains of resources besides the one based on OpenROADM. This first demonstration of FCAPS capabilities in OpenROADM transport network is likely to be of interest to other network operators and equipment vendors.

## References

1. ITU-T, "Recommendation M.3400 TMN Management Functions," https://www.itu.int/rec/T-REC-M.3400-200002-I/en (2000). Last retrieved 2/16/2021.
2. R. Casellas *et al.*, "Control, management, and orchestration of optical networks: Evolution, trends, and challenges," J. Light. Technol. **36**, 1390–1402 (2018 April).
3. B. Mirkhanzadeh *et al.*, "Demonstration of Joint Operation across OpenROADM Metro Network, OpenFlow Packet Domain, and OpenStack Compute Domain," Opt. Fiber Commun. Conf. p. W3C.3 (2020).
4. Neo4j, Inc., "neo4j Open Source Graph Database," https://github.com/neo4j (2021). Last retrieved 2/16/2021.
5. The GraphQL Foundation, "GraphQL A Query Language," https://github.com/graphql/foundation (2021). Last retrieved 2/16/2021.
6. Elastisearch B.V., "ELK Elastic Stack - Elastisearch, Logstash, Kibana data gathering, searching, visualization," https://github.com/elastic (2021). Last retrieved 2/16/2021.
7. Apache Software Foundation, "Kafka Stream Processing Software Platform," https://github.com/apache/kafka (2021). Last retrieved 2/16/2021.
8. The Linux Foundation, "Prometheus Open source systems monitoring and alerting toolkit," https://github.com/prometheus/prometheus (2021). Last retrieved 2/16/2021.
9. InfluxData, Inc., "InfluxDB Open Source Time Series Platform," https://github.com/influxdata/influxdb (2021). Last retrieved 2/16/2021.
10. Grafana Labs, "Grafana - query, visualize, alert on metrics and logs," https://github.com/grafana/grafana (2021). Last retrieved 2/16/2021.