Optical Network Control & Management Plane Evolution - A Large Datacenter Operator Perspective

Eric Breverman, Nancy El-Sakkary, Tad Hofmeister, Anees Shaikh, Vijay Vusirikala
Google Inc. 1600 Amphitheatre Parkway, Mountain View, CA 94043 USA
vijayvusiri@google.com

Abstract: Legacy management technologies and concepts are a major blocker to efficiently building and operating a large scale optical network. We provide an overview of new, modern device management technologies and discuss deployment and operational efficiencies that they enable.

1. Introduction

Optical networking equipment is ready for a drastic change with respect to management technologies and concepts. Until recently, most platforms were designed for manual human operation[1, 2]. Not only is it inefficient to continually hire more people as a network grows, but it is also increasingly challenging to find enough skilled engineers to manually operate a network. It has become obvious that automation is the way to achieve unprecedented scale in operating a network and speed in building a network. In short, only machines should interact with networking devices. This has driven the requirements for the key device building blocks of the ideal management solution--a modern API, vendor agnostic modeling, and declarative configuration.

2. Legacy Management

Historically, optical platform management was built for human interaction first (GUI, CLI, etc.). The platforms offered some quasi machine-to-machine communication, but these were typically old technologies (TL1 in 1984, SNMP in 1988)[3]. Most operators' attempts to automate their networks relied on doing their best to have automation interact with legacy technologies or a CLI. Automation was forced to deal with issues like user session management and parsing long strings into structured information. In addition, each new software release could adversely affect the code if string formats changed.

As the interaction was built for humans first, many unnecessary steps were required to perform an action. For example, performing a loopback is a multi-step process that often required putting a hierarchy of interfaces into a maintenance state before applying the loopback. Another example is that creating an entity often required several serial actions to create parent entities and put them in required states. The challenge is that each action is dependent on the current device state and the result of the previous action. This greatly increased the number of possible errors, created a lot of software complexity, and added no benefit.

Scaling on top of these foundations was also a major challenge. SNMP, for example, is not designed to be able to send *all* of the telemetry on a device northbound in near real time. Tradeoffs often had to be made between how fast and how much information was desired. In some scenarios, we have greatly exacerbated network outages by sending too many SNMP requests and crashing control planes so devices could not recover from external failures.

3. Requirements of a Modern Management Solution

The following design principles are important to successfully achieve the level of automation desired within the network.

- The transport mechanism for communicating to a device must be a true machine-to-machine protocol. It must be fast, efficient, highly structured, support streaming, and support advanced security mechanisms.
- The API (Application Programming Interface) must be standardized across vendors.

- Data modeling for the configuration and the telemetry must be standardized across vendors.
- An entire intended state device configuration must be able to be sent to the device and the device must handle all of the necessary transactions required to achieve the intended state.

4. Modern API

While there are several choices of Remote Procedure Call (RPC) frameworks available today, we advocate the use of gRPC for several reasons.

The most practical reason is the tight coupling and guaranteed contract of the service to a predefined protobuf definition. The benefit is that the services, requests and responses can be defined externally to the vendor. A gRPC service must conform to its protobuf definition, which allows vendor agnostic API's to be defined. As long as the vendor commits to using a given protobuf definition the resulting API is guaranteed. This is a critical architectural component to drive vendor agnostic API's.

Two standard protobuf projects have been defined through the OpenConfig community and are implemented on multiple vendor platforms today:

- GNMI (gRPC Network Management Interface) defines three RPC's for interacting with a device: Set, Get, and Subscribe. Set makes configuration changes to the device. Get performs a one-time retrieval of data from the device. Subscribe initiates streaming telemetry for a defined set of data.
- GNOI (gRPC Network Operations Interface) defines a set of RPC's for performing temporal actions on a device (Reboot, Cert Rotation, File Transfer).

Another key aspect of a modern API is streaming telemetry. Streaming telemetry is incredibly valuable in providing a simple architecture to enable sending near real time telemetry data. gRPC supports streaming natively and GNMI requests can specify to send information ON_CHANGE (event driven) or SAMPLE (regular, periodic streams). This powerful capability allows the device's software implementation to be written as a unidirectional architecture to efficiently stream *all* information on the device over one channel.

Lastly, security is heavily integrated with gRPC. Several features are available depending on operator requirements. Some of these features include SSL/TLS encryption, mutual authentication, credential-based authentication, and token-based authentication (i.e. OAuth2).

5. Vendor Agnostic Modeling

In a multi-vendor and multi-platform environment, differences between how each device is modeled creates tremendous downstream complexity. With telemetry, downstream services cannot simply ingest a given variable. They must first perform a lookup to see what the desired variable name is and then perform a transformation to a common variable type or common units. Because there may be hundreds of different variables collected from a given device, significant resources are required to overcome this.

With vendor agnostic modeling, variables and variable types are predefined allowing standardization between vendors. This allows huge simplification, allowing software to focus on its core scope as opposed to organizing vendor unique data. Vendor agnostic modeling and implementation also greatly simplifies new product introduction, regression testing, and vendor interoperability testing.

OpenConfig has proven to be an ideal data model for optical devices and it has been implemented on multiple vendor platforms. OpenConfig is an open source project which has allowed operators to define generic models that can be used across all networking gear. Along with the models, OpenConfig has built an ecosystem of libraries to easily enable using the models within code (i.e. pyangbind, ygot).

6. Declarative Configuration

A key need of next generation management is the ability to model the intended state network and push this model to individual devices as configuration. This is referred to as intent driven configuration.

The intended state is the source of truth for the network and is used by many other systems. If it is incorrect, many downstream systems will break. For example, you cannot troubleshoot a circuit outage if you do not know the topology or how the circuit is configured, so the network should reflect the intended state as quickly as possible. This is where declarative configuration plays a key role.

Declarative configuration allows a full configuration to be sent to a device and applied atomically. The full configuration includes everything on the device from management configuration, service configuration, admin states, loopback states, etc. At any time a full configuration may be pushed to a device regardless of its current state or configuration.

In an intent driven world, statefulness and intermediate steps should be avoided. For example, if modulation needs to be changed, a single intent update request with the new modulation, new client ports, etc. would be created. This is translated to a declarative configuration which would be pushed down to the device in one step. Another example is loopbacks. If a loopback needs to be enabled, it should be one change. If it is a multi-step process to move the port to a maintenance state, then enable a loopback, this requires a stateful system, which should be avoided.

Sequencing steps that are unique to each vendor to go from the current to the intended state adds unnecessary complexity. This has proven to be untenable as each vendor has many different tasks, each requiring different sequencing. This is greatly compounded in a multi-vendor network. We expect each vendor to know their platform best and allowing the vendor to implement this sequencing into their platform's software creates simplicity for operators.

7. Conclusion

Over the last few years an industry wide evolution has taken place to modernize optical platform management. Supporting an RPC framework, vendor agnostic modeling, and declarative configuration have enabled a modern approach to managing an optical network. This allows automation to permeate every action that needs to happen in the optical network (diagnostics, software rollouts, service activation, traffic rerouting, etc.). The result provides unprecedented scale in building and operating a network. Many of these requirements have been implemented on multiple platforms[4, 5, 6], ushering in an exciting era in managing optical networks.

References

1. O. F. Yilmaz, S. St-Laurent, and M. Mitchell, "Automated Management and Control of a Multi-Vendor Disaggregated Network at the L0 Layer," in Optical Fiber Communication Conference, OSA Technical Digest (online) (Optical Society of America, 2018), paper Tu3D.9.
2. A. Shaikh, T. Hofmeister, V. Dangui, and V. Vusirikala, "Vendor-neutral Network Representations for Transport SDN," in Optical Fiber Communication Conference, OSA Technical Digest (online) (Optical Society of America, 2016), paper Th4G.3.
3. Mani Subramanian, Network Management: Principle and Practice, 1st ed. Reading, MA: Addision-Wesley, Jan 2000.
4. Anees Shaikh, OpenConfig - progress toward vendor-neutral network management, Oct 2017, https://pc.nanog.org/static/published/meetings/NANOG71/1535/20171004_Shaikh_Lightning_Talk_Openconfig_v1.pdf.
5. Ciena - WaveServer AI Datasheet, 2018, https://media.ciena.com/documents/Waveserver-Ai-DS.pdf

6. Cisco - Data Models Configuration Guide for Cisco NCS 1001, March 2018, https://www.cisco.com/c/en/us/td/docs/optical/ncs1001/b_Datamodels_cg_ncs1001/b_Datamodels_cg_ncs1001_chapter_011.html