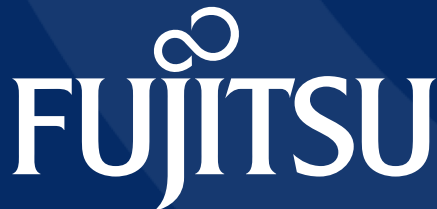




# Enabling Open and Disaggregated Transport Network with Modularized Network Gears Using Open Standards and Common Data Models

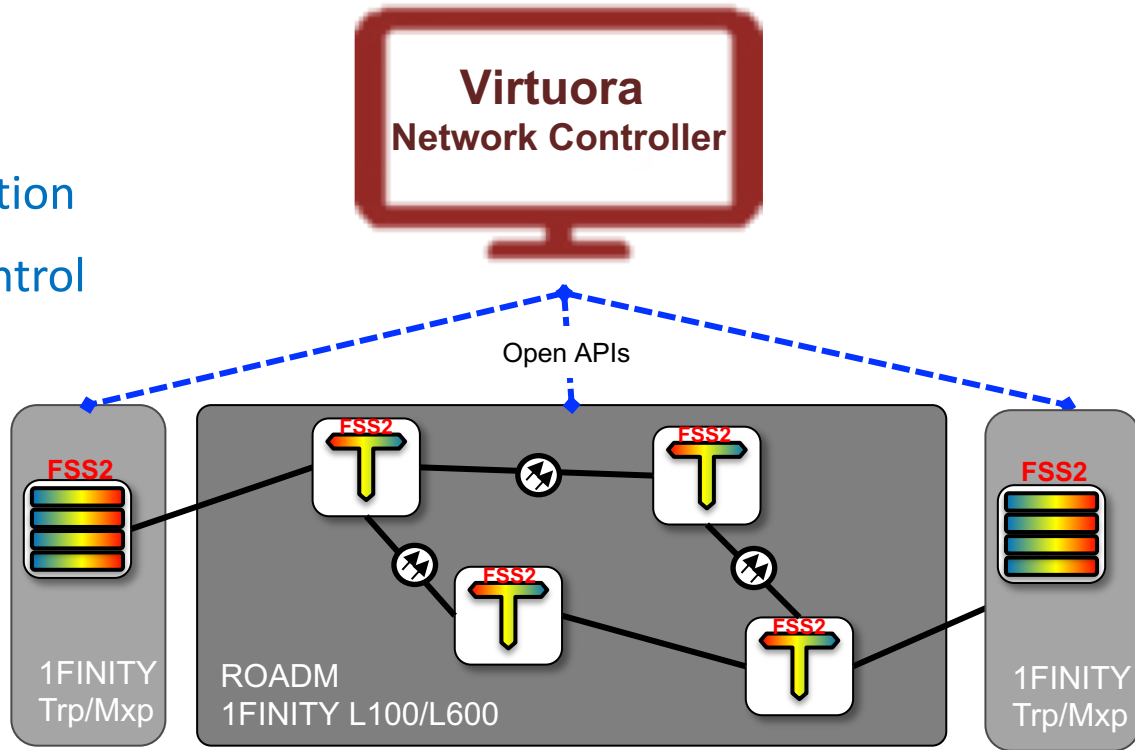
Yanbing Li, Calvin Wan

Fujitsu Network Communications, Inc.



# Open Optical Transport

- Vendor API -> Open API (e.g. OpenConfig, Open ROADM)
- Vendor lock-in boxes → ROADM/transponder disaggregation
- Hardware control -> software control
- Static -> dynamic network
- Closed-loop nodal and network automation
- Analytics
- Cloud



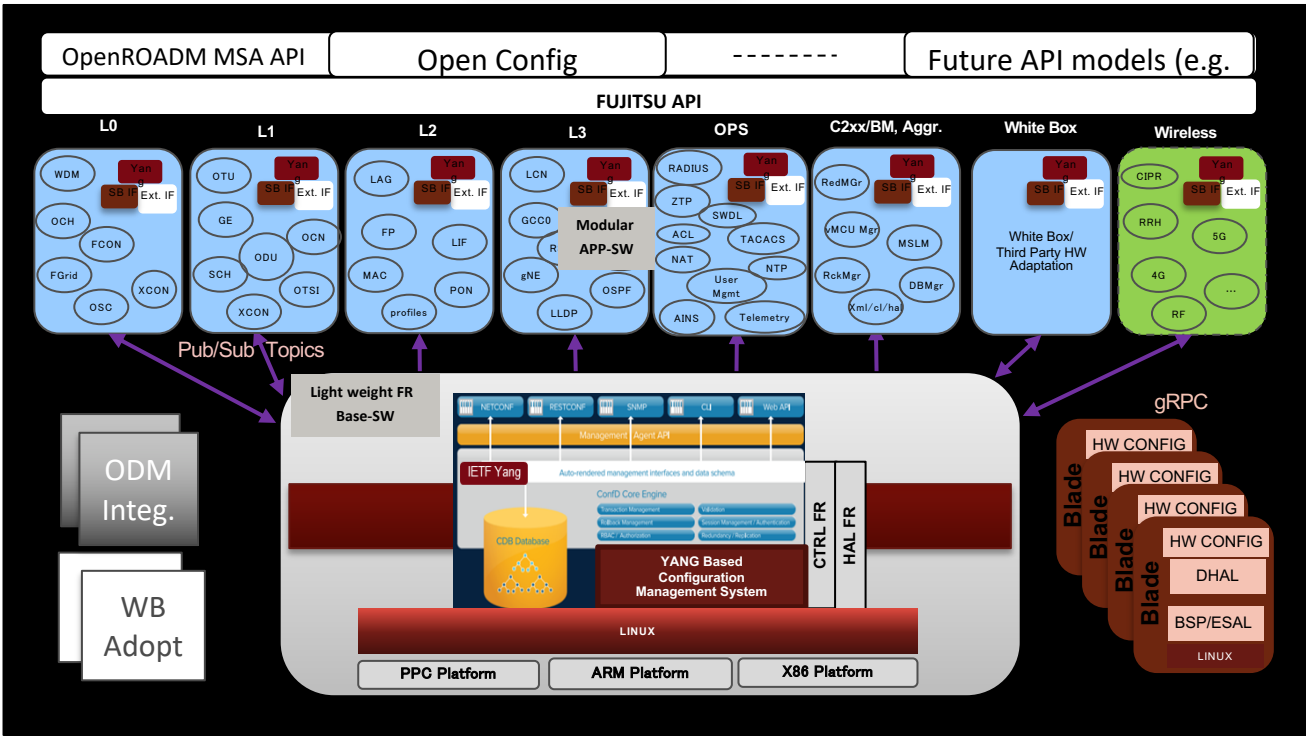
# FUJITSU System Software Version 2

- **Next Generation Carrier Grade software package from Fujitsu providing the latest Open Management interfaces with built in extensibility.**
  - Providing the same look and feel across all Products.
- **FSS2 is built on top of Linux and open source**
- **Running on many platforms**
  - Fujitsu Developed Products (1FINITY)
  - ODM Vendor hardware
  - Whitebox hardware
- **Field proven and hardened through deployment by many of the world's leading service providers.**



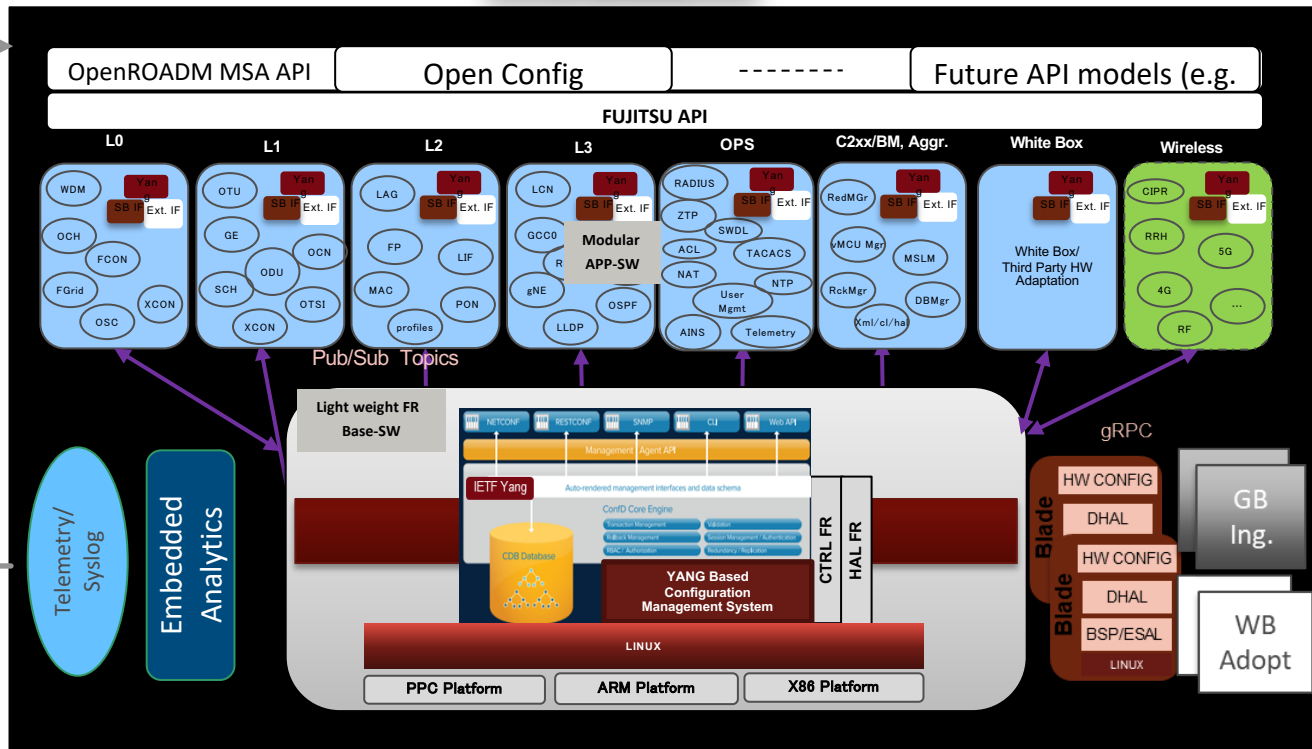
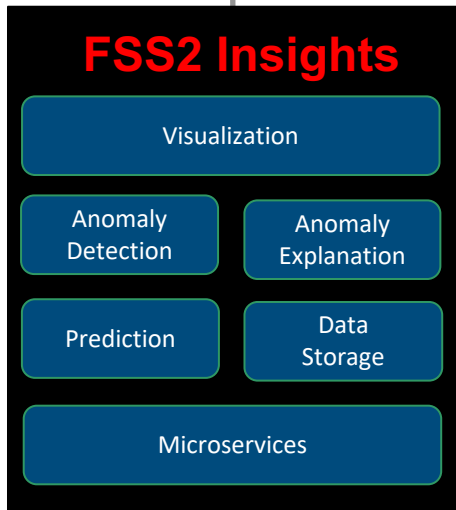
# FSS2

- **Modular design:** Base-SW, Blade-SW, APP-SW w/ BC NB, SB & inter-module APIs
- **Multiple data models ready:** IETF Yang, OpenROADM, OpenConfig, easy extendable to future emerging models
- **Model driven network interfaces:** Netconf, Restconf, gNMI, SNMP, WebGUI, CLI
- **Rich features:** 60+ OAMP operations & services. E.g. ZTP, Telemetry, Encryption
- **Eco-system ready:** easy greybox/whitebox adoptions and/or integration



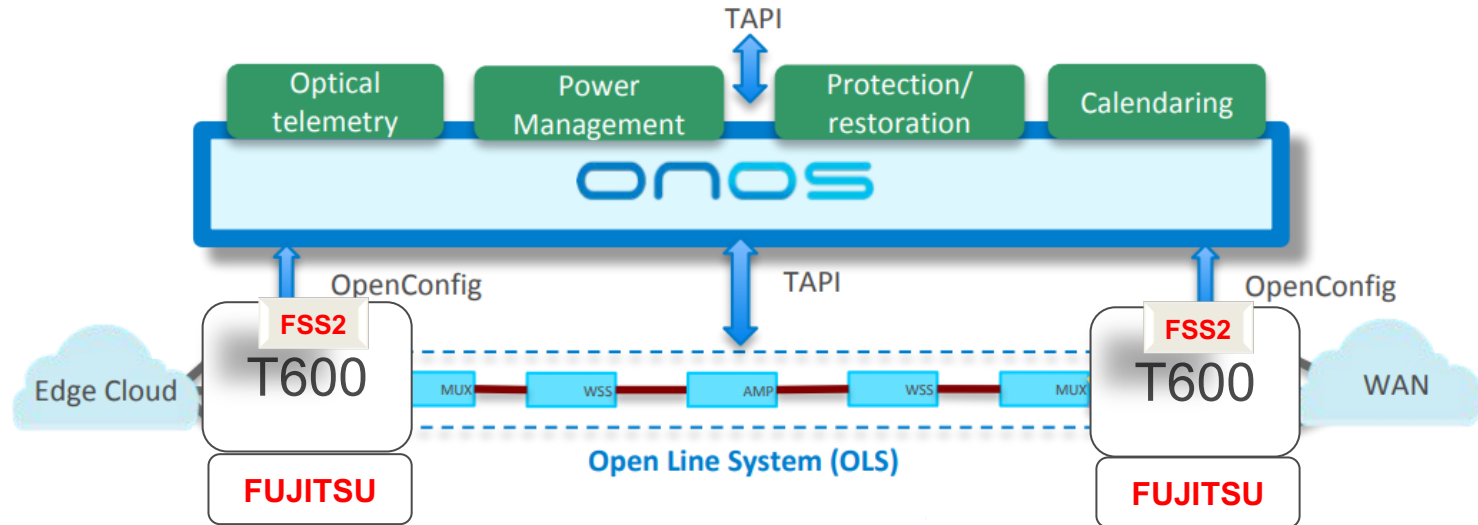
# Closed-loop Automation

**FSS2**



# FUJITSU's Contribution to ODTN Reference Model (Phase 1.0)

- Integrating T600 with ONOS via OpenConfig and NETCONF

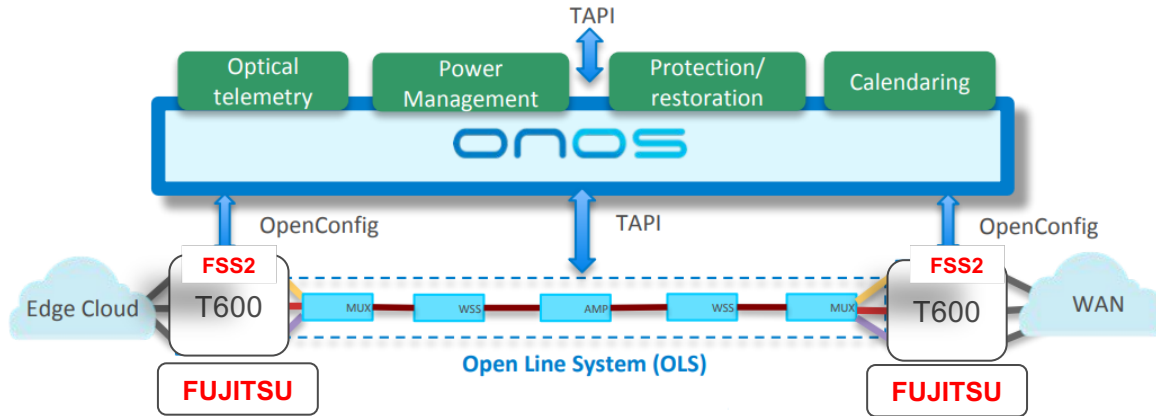


# Phase 1.0 Fujitsu Contribution Details

- T600-OC and ONOS integration per ODTN phase 1.0.
  - Transponder discovery:
    - Provide a way for OSS/BSS or Operator to send JSON with transponder endpoint details to ONOS
    - ONOS Initial reach out and OpenConfig request topology request
    - Transponder returns device information and ports
    - ONOS stores transponders device and ports in distributed store
  - Transponder provisioning:
    - ONOS create cross-connect (enable/disable traffic)
  - Identified and implemented a few of the missing pieces in ONOS gNMI:
    - Mutual TLS authentication (includes certificate exchange)
    - Username/password session per channel
- The 1FINITY T600 driver code is merged with ONOS upstream repository on GitHub
  - <https://github.com/opennetworkinglab/onos/commit/dee7e595a9950f9d87a2559e2abe19bf681b9eca>

# FUJITSU Contributing to ODTN Reference Model (Phase 1.5)

## Integrating 1FINITY T600 with ONOS via OpenConfig and NETCONF

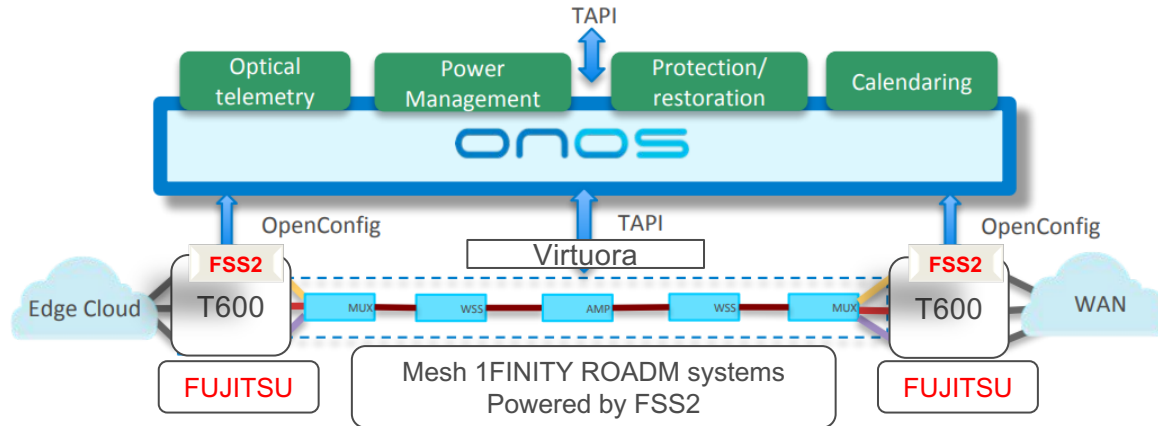


- FUJITSU transponder interop with an OLS for OTDN reference model
  - Transponder integration with ONOS and an OLS
  - NB OpenConfig for transponder (or per ODTN requirement)
  - Wavelength control – set a frequency on an optical channel



# FUJITSU Contributing to ODTN Reference Model (Future)

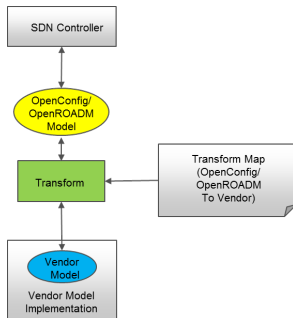
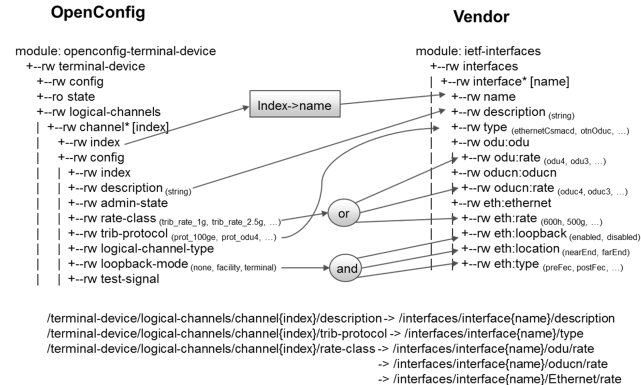
- Disaggregating Transponders from OLS



- Complete FUJITSU solution for OTDN reference model
  - Transponder, OLS
  - Northbound TAPI through Virtuora for OLS
  - NB OpenConfig for transponder (or per ODTN requirement)

# FSS2 Yang Model Transformation

- A challenge facing vendors and service providers is the ability to support a growing number of device models with minimum cost and fast time to market
- Model transformation is a software solution that
  - Enables a single product line to support multiple open models
  - Reduces the software development cost and time to market



## Transform map

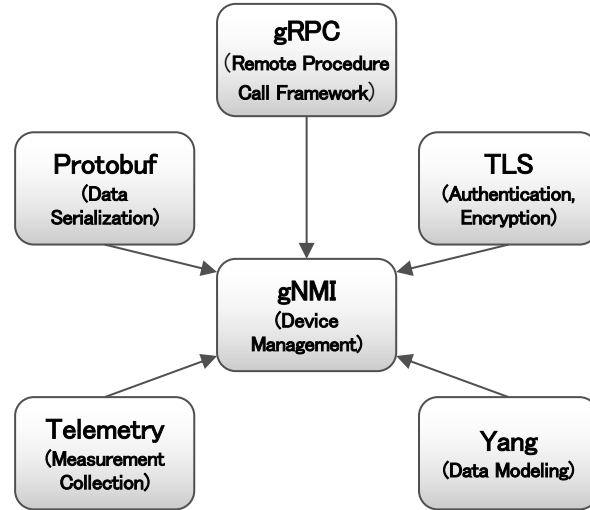
- Map attributes between open model and vendor model
- Specify transformation rules to transform open model attributes to vendor model attributes and vice versa

## Transform

- Open model agnostic
- Transform attributes based on rules specified in the transform map

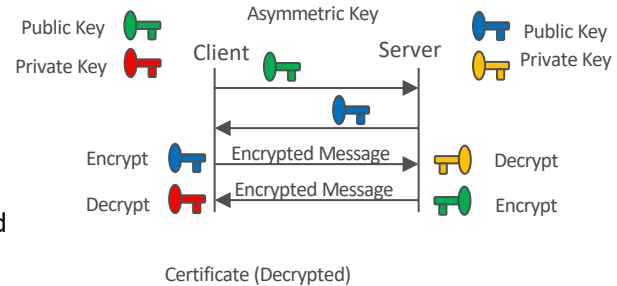
# gRPC Network Management Interface (gNMI)

- Support device configuration and telemetry control
- Use gRPC framework to define services
- Services
  - Capabilities – client to discover target capabilities (model, encoding, etc)
  - Get – client to retrieve data from target
  - Set – client to modify data on target
  - Subscribe – client to request target to send data
- Send mode
  - STREAM – target stream data to client
  - ONCE – target send data once to client
  - POLL – target send data upon client's poll request
- Subscription mode
  - ON\_CHANGE – send when element changes value
  - SAMPLE – send periodically
  - TARGET\_DEFINED – target defines ON\_CHANGE or SAMPLE for each element



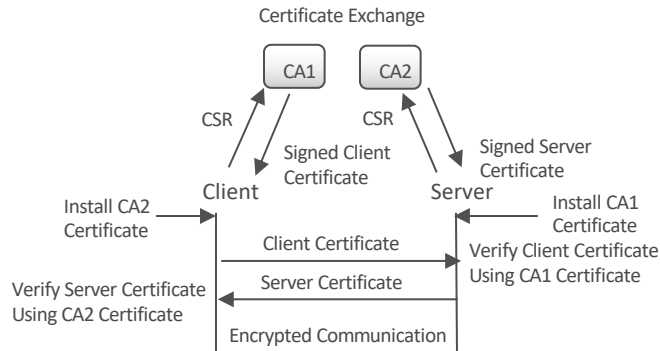
# Transport Layer Security (TLS)

- IETF standard to provide end-to-end communication security over networks
- Provide authentication and encryption via certificate
- Certificate authority (CA) – verify and sign client/server certificate
- Certificate signature request (CSR) – TLS client/server send CSR to CA to sign
- Certificate
  - CA-certificate – created by CA. Used to verify TLS client/server certificate
  - Client/server certificate – created by TLS client/server. Signed by CA



Certificate (Decrypted)

Certificate:  
 Data:  
 Version: 1 (0x0)  
 Serial Number: 1 (0x1)  
 Signature Algorithm: sha256WithRSAEncryption  
 Issuer: C=SP, ST=Spain, L=Valdepenas, O=Test, OU=Test, CN=Root CA  
 Validity  
 Not Before: Aug 22 15:43:50 2019 GMT  
 Not After : Aug 21 15:43:50 2020 GMT  
 Subject: C=US, ST=Texas, L=Dallas, O=Test, OU=Client, CN=www.example.com  
 Subject Public Key Info:  
 Public Key Algorithm: rsaEncryption  
 Public-Key: (4096 bit)  
 Modulus:  
 00:c3:ad:a8:ab:48:12:1b:54:57:0a:7e:bb:d5:c7:  
 ...  
 20:93:3b  
 Exponent: 65537 (0x10001)  
 Signature Algorithm: sha256WithRSAEncryption  
 a2:33:07:a2:14:7b:ea:d5:48:10:48:d4:54:07:4e:05:84:c7:  
 ...  
 3c:10:cb:ba:6f:6a:b6:d8



# Protocol Buffers (Protobuf)

- Efficient mechanism to serialize structured data
- Protobuf message is defined in a .proto file
- Message definition is language-neutral
- Message on the wire - stream of key-value pairs
- Backward compatible - new fields can be added to existing message without affecting existing software. New fields can simply be ignored by existing software
- Protobuf compiler generates stubs in many programming languages to read and write message fields
- Need protobuf definition to decode message

<https://github.com/openconfig/gnmi/blob/master/proto/gnmi/gnmi.proto>

```
...
message Notification {
  int64 timestamp = 1;    // Timestamp in nanoseconds since Epoch.
  Path prefix = 2;       // Prefix used for paths in the message.
  // An alias for the path specified in the prefix field.
  // Reference: gNMI Specification Section 2.4.2
  string alias = 3;
  repeated Update update = 4; // Data elements that have changed values.
  repeated Path delete = 5;  // Data elements that have been deleted.
  // This notification contains a set of paths that are always updated together
  // referenced by a globally unique prefix.
  bool atomic = 6;
}
...
```

# gRPC Remote Procedure Calls

- Open source remote procedure call framework
- gRPC client can invoke methods on remote gRPC server
- Support unary RPC, server streaming RPC, client streaming RPC, bidirectional streaming RPC
- Client and server can be written in different programming languages
- Use protobuf to define services and messages
- Support authentication and encryption

gRPC Layers

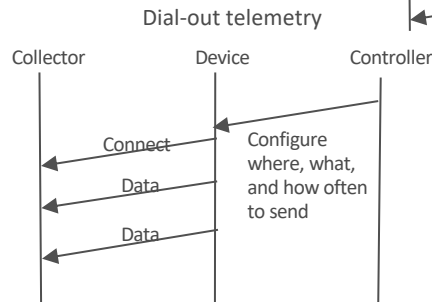
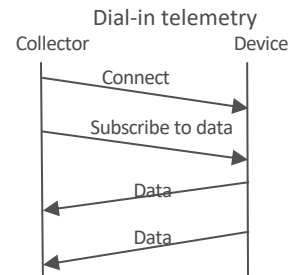
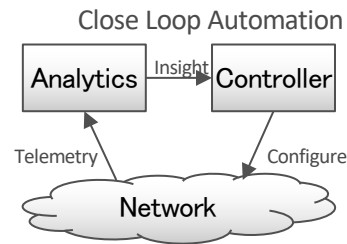


<https://github.com/openconfig/gnmi/blob/master/proto/gnmi/gnmi.proto>

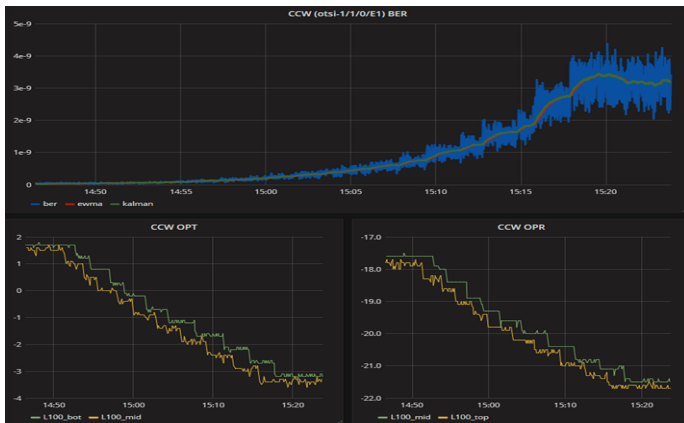
```
...
service gNMI {
  // Capabilities allows the client to retrieve the set of capabilities that
  // is supported by the target. This allows the target to validate the
  // service version that is implemented and retrieve the set of models that
  // the target supports. The models can then be specified in subsequent RPCs
  // to restrict the set of data that is utilized.
  // Reference: gNMI Specification Section 3.2
  rpc Capabilities(CapabilityRequest) returns (CapabilityResponse);
  // Retrieve a snapshot of data from the target. A Get RPC requests that the
  // target snapshots a subset of the data tree as specified by the paths
  // included in the message and serializes this to be returned to the
  // client using the specified encoding.
  // Reference: gNMI Specification Section 3.3
  rpc Get(GetRequest) returns (GetResponse);
  // Set allows the client to modify the state of data on the target. The
  // paths to modified along with the new values that the client wishes
  // to set the value to.
  // Reference: gNMI Specification Section 3.4
  rpc Set(SetRequest) returns (SetResponse);
  // Subscribe allows a client to request the target to send it values
  // of particular paths within the data tree. These values may be streamed
  // at a particular cadence (STREAM), sent one off on a long-lived channel
  // (POLL), or sent as a one-off retrieval (ONCE).
  // Reference: gNMI Specification Section 3.5
  rpc Subscribe(stream SubscribeRequest) returns (stream SubscribeResponse);
}
...
```

# Telemetry

- Mechanism to transmit measurements from a device to remote servers
- Data – PM, alarm, log, cpu usage, memory usage, etc
- Use cases – monitoring, anomaly detection, failure prediction, capacity planning, etc



Optical Power/Bit Error Rate Analysis



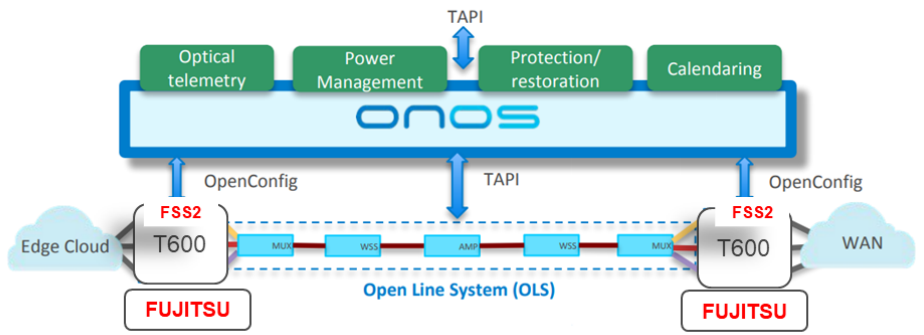
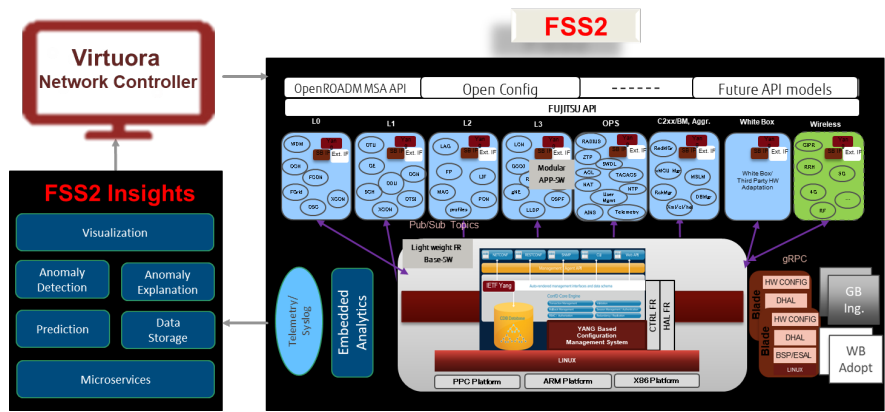
# Key Takeaways

Continual collaboration & contribution with

- ONF and partners on OTDN project
- open eco-system using Virtuora, 1FINITY, FSS2 platforms

## FUJITSU Contributing to ODTN Reference Model (Phase 1.5)

Integrating 1FINITY T600 with ONOS via OpenConfig and NETCONF




- FUJITSU transponder interop with an OLS for OTDN reference model
  - Transponder integration with ONOS and an OLS
  - NB OpenConfig for transponder (or per ODTN requirement)
  - Wavelength control – set a frequency on an optical channel







  
FUJITSU

Thank You

Follow Up Links:

[Yanbing.li@us.fujitsu.com](mailto:Yanbing.li@us.fujitsu.com)

[Calvin.wan@us.fujitsu.com](mailto:Calvin.wan@us.fujitsu.com)