First demonstration of YANG push notifications in Open Terminals

Arturo Mayoral, Victor Lopez, Juan Pedro Fernandez Palacios
Telefonica I+D
Madrid, Spain
arturo.mayoral@telefonica.com

Yuan Yufeng, Zhou Lifen, Hu Wenjun, Zhang Mingfeng, Yan Changlong
*FiberHome*Wuhan, China
yuanyf@fiberhome.com

***Abstract*—Real time information retrieval enables stay in sync with the network devices' status. Solutions based on polling, are not optimal in network deployments, the devices have to interrupt their processes to send the information, which is received with some delay and may be critical for many applications. Yang push notifications define a mechanism to subscribe and to push for Yang datastore updates, improving the performance of previous technologies. This work demonstrates for the first time the implementation of YANG push notifications on an Open Terminal based on OpenConfig models.**

Keywords—Software Defined Networking; Telemetry; Disaggregation; YANG Notifications; Open Networking

I. INTRODUCTION

Network operators provides transport services to their customers thanks to systems and solutions developed by a single network vendor. Closed solutions delay the introduction of new functionalities as well as limits and innovation. Operators have been seeking for solutions for many years, to introduce automation in the provisioning and management of services in multi-vendor, multi-layer [1] and multi-domain networks [2], by betting on the development of standard interfaces. Software Defined Networking (SDN) was conceived as an architectural solution designed to centralize the control and management of the network into a software-based controller. SDN has been evolving towards the standardization of the interfaces capable to realize itself. NETCONF [3] is a management protocol for network elements based on datastores defined in YANG, and RESTconf [4] is its' light counterpart, adopted to standardize the structure of the REST APIs between the SDN controllers, defined according to the underlying YANG datastores. These two protocols and the YANG modelling language is the present of SDN solutions nowadays.

Network disaggregation is a relevant concept for all the industry including network operators [5]. Fig. 1 illustrates the three main alternatives when considering disaggregation in optical networks. Each architecture differs in the level of disaggregation. An aggregated architecture consists of a network where all elements of the line system belong to the same vendor and operate as an end-to-end entity (Fig. 1-a). Fig 1-b shows a partially disaggregated scenario, which have two elements: Open Terminals (OT) and Open Line System (OLS). The OTs are the transponders with a standard interface and the OLS is composed by the WSSs, wavelength multiplexers, optical amplifiers, etc. Fig 1-c depicts a fully disaggregated scenario decouples. This alternative separates all elements in a line system and they operate as a stand-alone entities.

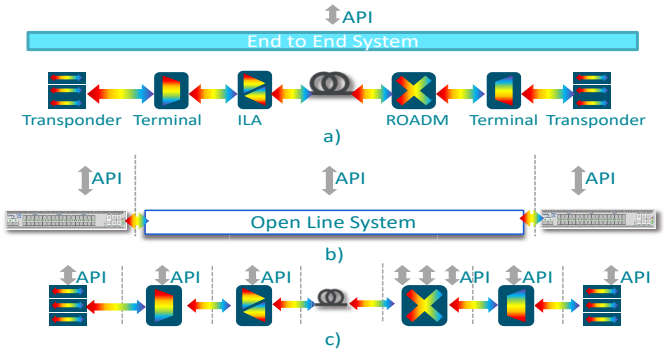

Fig. 1. Disaggregated Network Architectures: a) Aggregated, b) Partially Disaggregated and c) Fully Disaggregated [3].

Current solutions for real time information retrieval, used for various network management purposes service's performance monitoring, network reconciliation or assurance are based on polling, strategy which may be suboptimal in some cases. Polling incurs significant latency, both because the event might occur at any moment in between of the client polling period and because the device introduces some delay during the system request processing. This latency precludes many real-time of applications which relay on instant information arrival to operate under this mechanism, which may even saturate the DCN network when the amount of data collected from all network elements is the greatest. Another scenario are applications that track datastore configuration changes through notifications. To get the event, many remote polling cycles are done without any change detected and wasting resources in the network, devices, and applications. This is clear in scenarios where the changes occur only infrequently.

As stated before, NETCONF is the main protocol for network element configuration. However, it was defined without any support for notifications. With RFC5277 [6], NETCONF event notifications were defined as a method to send asynchronous messages within a NETCONF implementation. However, the notifications must be statically defined within the YANG model and there is not a criteria to define periodic or on-change notifications. YANG notifications overcome the limitations of NETCONF event notifications. gRPC Remote Procedure Calls (RPC) [7] is a high-performance protocol originally defined for telemetry. It is very efficient for real time information retrieval, authors in [8, 9] worked on gRPC for flexigrid and Spatial Division Multiplexing (SDM) scenarios. This work is focus on demonstration YANG push notifications

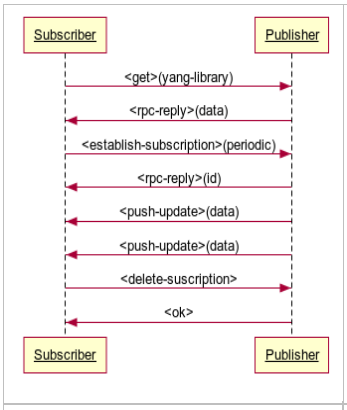

Fig. 2. Periodic Subscription

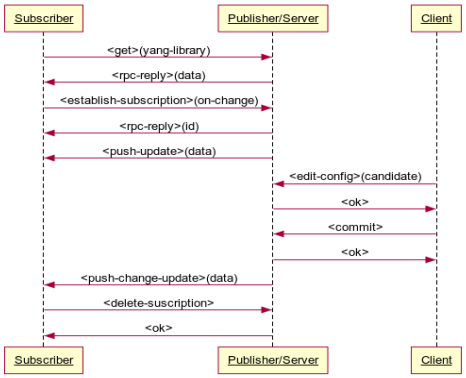

Fig. 3. On-change Subscription

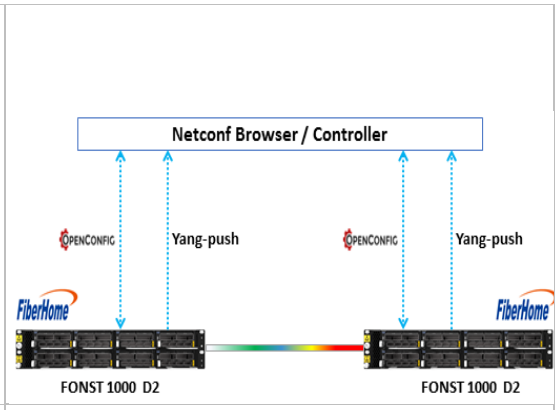

Fig. 4. Schematic of the lab set-up

(RFC8641 – [10]) as it overcomes some limitations of RFC5277 and NETCONF is already supported by many devices in operators networks.

II. YANG NOTIFICATIONS

RFC8639 [11] defines a YANG model and the mechanisms allowing the subscriptions to a publisher's event streams for any transport protocol. RFC8639 defines the mechanism a subscriber needs to use to request and to receive an information feed from a YANG datastore server or publisher. In this academic exercise, we are positioning the SDN controller as the network elements' subscriber, while the publisher in a disaggregated network scenario would be the Open Terminals or the OLS. This new protocol extends the work in RFC5277 which had some limitations, e.g., the subscriber could not modify or dynamically change the subscriptions, adapting to a change on the network conditions. There was not a negotiation phase in the parameters and the operation counters were not accessible. Additionally, RFC8639 is transport protocol-independent so the mechanism can be used not just with NETCONF but with RESTConf. YANG push defined in RFC8641 [10] extends the subscription model from RFC8639 [11] with capabilities that allow subscribers to define the triggers to retrieve updates and include filters.

```
Command XML
<establish-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"
                        xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push"
                        xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <yp:datastore xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
ds:operational
</yp:datastore>
  <yp:datastore-subtree-filter>
    <components xmlns="http://openconfig.net/yang/platform">
      <component>
        <linecard xmlns="http://openconfig.net/yang/platform/linecard"/>
      </component>
    </components>
  </yp:datastore-subtree-filter>
  <yp:periodic>
    <yp:period>500</yp:period>
  </yp:periodic>
</establish-subscription>
```

Fig. 5. Establish-subscription message for a periodic subscription

```
<?xml version="1.0" encoding="utf-8"?>
<rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
           xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
           message-id="5">
  <id xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications">1</id>
</rpc-reply>
```

Fig. 6. Response to the establish-subscription message

Generated	Received	Notification
Tue, Jan, 21, 2020, 10:38:16 +0800	Tue, Jan, 21, 2020, 10:38:18 +0800	push-update
Tue, Jan, 21, 2020, 10:38:21 +0800	Tue, Jan, 21, 2020, 10:38:23 +0800	push-update
Tue, Jan, 21, 2020, 10:38:26 +0800	Tue, Jan, 21, 2020, 10:38:28 +0800	push-update
Tue, Jan, 21, 2020, 10:38:31 +0800	Tue, Jan, 21, 2020, 10:38:33 +0800	push-update
Tue, Jan, 21, 2020, 10:38:36 +0800	Tue, Jan, 21, 2020, 10:38:38 +0800	push-update
Tue, Jan, 21, 2020, 10:38:41 +0800	Tue, Jan, 21, 2020, 10:38:43 +0800	push-update
Tue, Jan, 21, 2020, 10:38:46 +0800	Tue, Jan, 21, 2020, 10:38:48 +0800	push-update
Tue, Jan, 21, 2020, 10:38:51 +0800	Tue, Jan, 21, 2020, 10:38:53 +0800	push-update
Tue, Jan, 21, 2020, 10:38:56 +0800	Tue, Jan, 21, 2020, 10:38:59 +0800	push-update

Fig. 7. Packets capture with the push-update notification

YANG push defines two subscription mechanisms, one is periodic subscription, the other is on-change subscription, the workflows of which are shown as Fig. 2 and Fig. 3. Periodic subscription is created when sending a *establish-subscription* message to the publisher including the period and the subtree we want to receive. Once the subscription request is accepted by the publisher, the publisher will send a *push-update message* with the subtree information periodically. Similarly, on-change subscription is created with a *establish-subscription message*, which includes the on-change parameter and the subtree we want to receive (Fig. 3). After the acceptance by the publisher, the publisher sends *push-update messages* every time that there is a change in the selected subtree. If there is a change in the configuration, the publisher sends *push-change-update messages* to the subscriber to know the reason of the modification.

III. EXPERIMENTAL DEMONSTRATION

A. Testbed Description

To carry out the experiments, two Open Terminals (FiberHome FONST 1000 D2) are deployed in the lab tests, shown as Fig. 4. They expose YANG push notifications using NETCONF as the transport protocol. To validate the syntax, NETCONF browser is used. It is a commonly used and user-friendly NETCONF client application, which could be considered as the controller retrieving the information from the Open Terminals. In this work we have considered OpenConfig [12] as the information model.

B. Periodic Notification

As introduced in the previous section, Yang push has two kinds of subscription, one is periodic subscription, the other is on-change subscription. To demonstrate that the device supports the Yang notifications, first, it is sent a <*establish-subscription*> RPC request to the device as depicted in Fig. 5. The message includes *subtree-filter* with the information to monitor. In this case, we monitor the linecard components. A periodic notification requires the period, which is 500 in this case,

defined in centiseconds, 5s. The device accepts the notification request as shown in Fig. 6. Once the notification is configured, the Open Terminal sends a *push update* notification every 5 seconds as shown in Fig. 7. Fig. 8 shows the leaves defined within the OpenConfig model for the linecard element.

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2020-01-21T02:38:16Z</eventTime>
  <push-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <id>1</id>
    <datastore-contents>
      <components xmlns="http://openconfig.net/yang/platform">
        <component>
          <linecard xmlns="http://openconfig.net/yang/platform/linecard">
            <state>
              <slot-id>1</slot-id>
            </state>
          </linecard>
          <name>LINECARD=/chassis=0/slot=1</name>
        </component>
        <component>
          <linecard xmlns="http://openconfig.net/yang/platform/linecard">
            <state>
              <slot-id>9</slot-id>
            </state>
          </linecard>
          <name>LINECARD=/chassis=0/slot=9</name>
        </component>
      </components>
    </datastore-contents>
  </push-update>
</notification>
```

Fig. 8. Detailed information in the notification for the periodic subscription

C. *On-change Notification*

Once periodic notifications are demonstrated, on-change notifications are described. Fig. 9 depicts the <*establish-subscription*> RPC request with the *on-change* parameter included. Moreover, the RPC include the subtree, which is the logical channels in this example. Fig. 10 illustrates the *push-change-update* notification, which is sent because there is a new logical channel created. The logical channel is a10G Ethernet service as shown in Fig. 11.

IV. Conclusions

Real time information retrieval allows to retrieve the network element status. This information is becoming more and more important to adapt the network to changes thanks to the deployment of SDN controllers. Network operators are migrating their optical networks to disaggregated architectures. his work demonstrates for the first time the support of YANG push notifications in Open Terminals. This capability is key to retrieve the transponder information in disaggregated scenarios and provide to the systems an end-to-end view of the optical network.

Acknowledgment

This work has been supported partially by EU H2020 project Metro-haul, grant no. 761727 (https://metro-haul.eu).

References

[1] A. Mayoral Lopez-de-Lerma, V. Lopez, M. López Bravo, D. García Montes, O. Gonzalez de Dios, A. Aguado, R. Szwedowski, K. Mrówka, F. Maques, Z. Stevkovski, D. Verchere, Q. Pham Van, L. Tancevski and J.P. Fernandez Palacios: Multi-layer service provisioning over resilient Software-Defined partially disaggregated networks, in Journal of Lightwave Technology.

[2] V. López, O. Gonzalez de Dios, L.M. Contreras, J. Foster, H. Silva, L. Blair, J. Marsella, T. Szyrkowiec, A. Autenrieth, C. Liou, A. Sasdasivarao, S. Syed, J. Sun, B. Rao, F. Zhang and J.P. Fernandez-Palacios: Demonstration of SDN orchestration in optical multi-vendor scenarios, in Proc. Optical Fiber Conference (OFC), Mar 2015

[3] R. Enns, Ed., Network Configuration Protocol (NETCONF), IETF RFC 6241, 2011.

[4] A. Bierman, M. Bjorklund, K. Watsen, RESTCONF protocol, IETF RFC 8040, 2017.

[5] Victor Lopez, Wataru Ishida, Arturo Mayoral, Takafumi Tanaka, Oscar Gonzalez de Dios, and Juan Pedro Fernandez-Palacios, "Enabling fully programmable transponder white boxes", J. Opt. Commun. Netw. 12, A214-A223 (2020).

[6] S. Chisholm and H. Trevino, IETF RFC5277 "NETCONF Event Notifications"

[7] gRPC. https://grpc.io/Accessed: 2019-02-14.

[8] F. Paolucci, A. Sgambelluri, M. Dallaglio, F. Cugini, and P. Castoldi, Demonstration of gRPC telemetry for soft failure detection in elastic optical networks, in Proc. of 2017 European Conference on Optical Communication (ECOC) (pp. 1-3). IEEE, 2017.

[9] R. Vilalta, N. Yoshikane, R. Casellas, R. Martínez, S. Beppu, D. Soma, S. Sumita, T. Tsuritani, I. Morita, R. Muñoz, GRPC-based SDN control and telemetry for soft-failure detection of spectral/spacial superchannels , in Proceedings of 45th EuroPean Conference on Optical Ommunication (ECOC), 2019.

[10] RFC 8641, A. Clemm and E. Voit "Subscription to YANG Notifications for Datastore Updates"

[11] RFC 8639, E. Voit, et al. "Subscription to YANG Notifications".

[12] OpenConfig, http://www.openconfig.net/


```
Command XML
<establish-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-notifications"
                        xmlns:yp="urn:ietf:params:xml:ns:yang:ietf-yang-push"
                        xmlns:netconf="urn:ietf:params:xml:ns:netconf:base:1.0">
  <yp:datastore xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">
ds:operational
</yp:datastore>
  <yp:datastore-subtree-filter>
    <terminal-device xmlns="http://openconfig.net/yang/terminal-device">
      <logical-channels>
        <channel/>
      </logical-channels>
    </terminal-device>
  </yp:datastore-subtree-filter>
  <yp:on-change/>
</establish-subscription>
```

Fig. 9. Establish-subscription message for a on-change subscription

```
/transport-types">oc-opt-types:TRIB_RATE_10G</rate-class>
ang/transport-types">oc-opt-types:PROT_10GE_WAN</trib-protocol>
g.net/yang/transport-types">oc-opt-types:PROT_ETHERNET</logical-channel-type
```

Fig. 11. Zoom in the detailed notification for the on-change subscription

```
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2020-01-21T02:43:28Z</eventTime>
  <push-change-update xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">
    <id>2</id>
    <datastore-changes>
      <yang-patch xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-patch">
        <patch-id>1</patch-id>
        <edit>
          <edit-id>1</edit-id>
          <operation>create</operation>
          <target>/oc-opt-term:terminal-device</target>
          <value>
            <terminal-device xmlns="http://openconfig.net/yang/terminal-device">
              <logical-channels>
                <channel>
                  <index>1001</index>
                  <config>
                    <index>1001</index>
                    <loopback-mode>NONE</loopback-mode>
                    <rate-class xmlns:oc-opt-types="http://openconfig.net/yang/tra
                    <trib-protocol xmlns:oc-opt-types="http://openconfig.net/yang/
                    <logical-channel-type xmlns:oc-opt-types="http://openconfig.ne
                  </config>
                </channel>
              </logical-channels>
            </terminal-device>
          </value>
        </edit>
      </yang-patch>
    </datastore-changes>
  </push-change-update>
</notification>
```

Fig. 10. Detailed information of the on-change subscription