Experimental Demonstration of Active and Passive Optical Networks Telemetry

Lluis Gifre[1], Jose-Luis Izquierdo-Zaragoza[2], Behnam Shariati[2], and Luis Velasco[2*]

[1] Universidad Autónoma de Madrid (UAM), Madrid, Spain
[2] Optical Communications Group (GCO), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
e-mail:lvelasco@ac.upc.edu

Abstract: A distributed architecture enabling active and passive optical network telemetry is presented; a YANG data model is used for remotely configuring monitoring devices for telemetry purposes. Experimental demonstration is carried out for failure localization.
© 2018 Optical Society of America
OCIS codes: (060.4250) Networks; (120.4800) Optical standards and testing

1. Introduction

Operators need as much information as possible about their network performance, based on measurements, so they can identify, isolate, and solve potential issues as quickly as possible to keep their networks running smoothly. In this regard, passive monitoring techniques are the most-common choice in optical networks, since they entail using non-invasive methods to obtain measurements; examples include measuring Bit Error Rate (BER) in optical transponders, the optical power in optical links, as well as acquiring the optical spectrum in the whole C-band in links using optical spectrum analyzers (OSAs). Nonetheless, some active monitoring techniques are also available at the optical layer; for instance, the authors in [1] proposed to use in-band Optical Supervisory Channel (OSC) devices to monitor BER of 100 Gb/s DP-QPSK lightpaths in intermediated points, i.e., not in the transponders. This technique consists in over-modulating the target lightpath with low-speed low-bandwidth OOK signal in an OSC_{TX} device at the ingress node. Then, the BER of the OOK signal is measured at intermediate locations by OSC_{RX} devices using low-speed components, and the BER of the lightpath is estimated by BER correlation curves obtained *a priori*.

Regarding architectures supporting monitoring, authors in [2] presented CASTOR, a distributed monitoring and data analytics architecture that consists of two main blocks: *i*) *hypernodes* are distributed agents close to network devices, and *ii*) a centralized data analytics module running in the control and management (C&M) plane together with the network SDN controller. Hypernodes collect data from network devices, aggregating them before being forwarded to the centralized data analytics module. Besides, local *knowledge discovery from data* (KDD) processes can be used to pre-process and aggregate data received from the network nodes. The centralized data analytics module includes a repository for monitoring data, as well as the operational databases (TED and LSP-DB) retrieved from the SDN controller, which entails having a complete view of the network.

In this paper, we experimentally demonstrate two failure localization methods based on the use of OSAs and OSCs, as proposed in [3]. CASTOR is extended to complete the network telemetry life-cycle. A novel YANG data model has been created to expose monitoring capabilities ("what", "where" and "how") from network devices to the C&M plane, as well as for configuration of monitoring procedures from the C&M plane.

2. Architecture, data model, and workflows

Fig. 1 overviews extended CASTOR architecture to support active and passive telemetry. Although other configurations are possible, without loss of generality, we assume that monitoring devices (i.e., OSAs and OSC_{TX} and OSC_{RX} devices) are installed in the optical nodes and thus, optical cross-connect (OXC) agents are the common element for controlling the node itself as well as monitoring devices, including telemetry.

CASTOR architecture has been extended for node control: hypernodes now include a *local config* module, collocated with the *local KDD* module for data collection, enabling control of network devices and supporting multiple network nodes. Two north/south-bound interfaces (NBI and SBI) connect the hypernode with both C&M and network nodes, respectively: *i*) IPFIX protocol is selected for monitoring because of its capabilities to define custom templates for specific purposes, and *ii*) RESTCONF (in the NBI) and gNMI (in the SBI) protocols based on a common YANG data models are used for configuration and event notifications.

Without entering into details, we define a YANG data model with two subtrees, one for network component (interfaces and LSPs) configuration and another for monitoring; note that the local configuration module focuses on the configuration subtree, whereas local KDD module focuses on the monitoring subtree. Network devices are assumed to feed a data store compliant with this model, e.g., by using an intermediate driver for message translation purposes, whereas dissemination toward hypernodes and SDN controller can be made through notifications during bootstrapping, upon updates, or by polling. Fig. 2 presents an example for a lightpath (L0_LSP): configuration parameters include, but are not limited to, modulation format, bit/baud-rate, and the allocated frequency slot specified by *n* and *m* parameters [4]. In the monitoring subtree, components include two

The research leading to these results has received funding from the EC through the METRO-HAUL project (G.A. nº 761727), from the Spanish MINECO TWINS project (TEC2017-90097-R), and from the Catalan Institution for Research and Advanced Studies (ICREA).

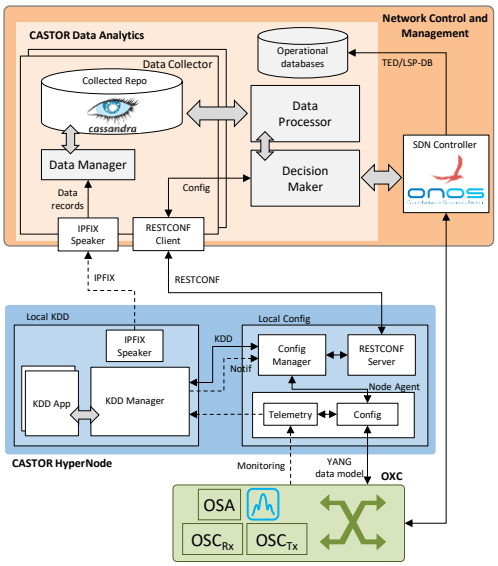

Fig. 1. CASTOR telemetry and data analytics manages active and passive monitoring devices installed in OXCs.

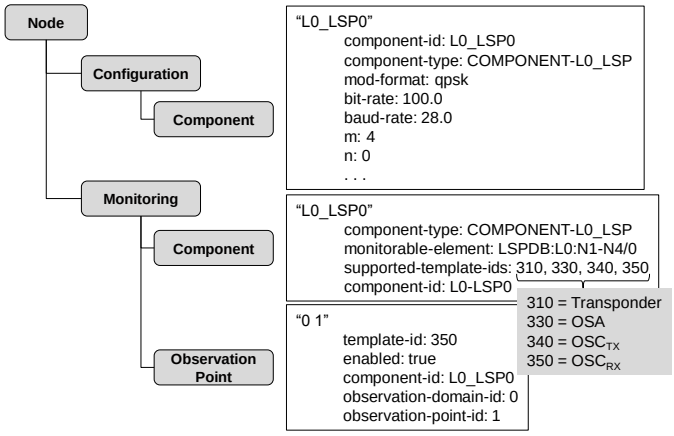

Fig. 2. YANG data model to support active and passive monitoring.

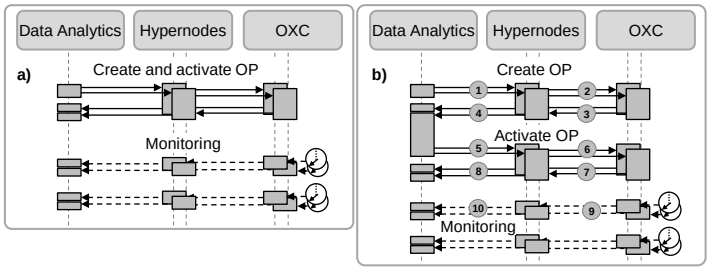

Fig. 3. OPs activation and monitoring workflows: a) for OSAs and b) for OSCs.

analytics in the C&M plane; and *ii*) *supported-template-ids* ("how"), relevant attributes: *i*) *monitorable-element* ("what"), which is used for correlation purposes with the operational databases in the SDN controller and by CASTOR data representing the set of monitoring methods that such component supports in the specific network node ("where"). Without loss of generality, we use IPFIX template identifiers as monitoring method identifiers: *i*) template 310, used to monitor BER/receiver power in the transponders; *ii*) template 330, used by OSAs to send sequences of tuples <frequency, optical power>; *iii*) template 340, used by OSC$_{TX}$ devices as heartbeat/keepalive; and *iv*) template 350, used by OSC$_{RX}$ devices to send estimated BER values. Although all the templates have been included in Fig. 2 for illustrative purposes, only those supported by the component in the network node will be advertised. Network devices are expected to allocate monitoring resources by translating tuples <component-id, template-id> into device-specific commands. Observation domain/point identifiers are internal identifiers used by KDD processes inside CASTOR hypernodes to isolate different datasets, and they are included in IPFIX messages. Multiple observation points (OPs) can be defined for a single component, each of them reporting monitored data formatted as per the selected supported template.

Fig. 3 presents the workflows to activate OPs in the network nodes aiming at monitoring a lightpath in all the nodes along its route using OSAs (Fig. 3a) and using OSCs (Fig. 3b). Both workflows are similar; OPs need to be created and activated in the nodes, so the centralized CASTOR data analytics module issues a message to every hypernode in charge of a node in the route of the lightpath. Once OPs are activated, data records start to be periodically generated by the monitoring devices and collected by the corresponding hypernode, which in turn processes them and forwards the aggregated counterparts of the collected data toward the data analytics module. The difference between workflows lies in the way creation and activation are performed. In the case of OSAs, the devices are already acquiring the whole C-band, hence creation and activation of OPs are performed in a single step. By contrast, when OSCs are selected for monitoring, one OSC$_{TX}$ module in the ingress node and one OSC$_{RX}$ module in the rest of the nodes along the route need to be reserved (a limited number of modules might exist in the nodes) so that OPs might not be activated unless OSC modules where successfully allocated.

3. Experimental demonstration

Fig. 4 presents an example of failure localization using active and passive monitoring techniques on the lightpath established from OXC-1 to OXC-7. As illustrated by the node architecture in Fig. 4a, pools of OSC$_{TX}$ and OSC$_{RX}$ are available in each OXC, whereas OSAs are installed in all outgoing optical links and acquire the whole C-band. Let us assume a filter shift problem in OXC-5 as a result of misconfiguration in a WSS.

Let us first demonstrate the case of using OSCs to localize the failure. In this case, one OSC$_{TX}$ needs to be allocated in OXC-1, while OSC$_{RX}$ needs to be allocated in every subsequent node along the lightpath (i.e., OXCs 2, 5, 3, and 7). Fig. 5 presents the messages captured for the defined OSC telemetry workflow, where messages have been grouped for the three sequential phases in the workflow: create OPs that entails reserving OSC devices in the nodes (messages 1-4), activate OSCs that involves configuring the OSC devices (5-8), and monitoring (9-10); note that for the sake of brevity, only messages exchanged with OXC-1 are shown, being those with the other OXCs equivalent. Since the hypernode might aggregate and transform data received from

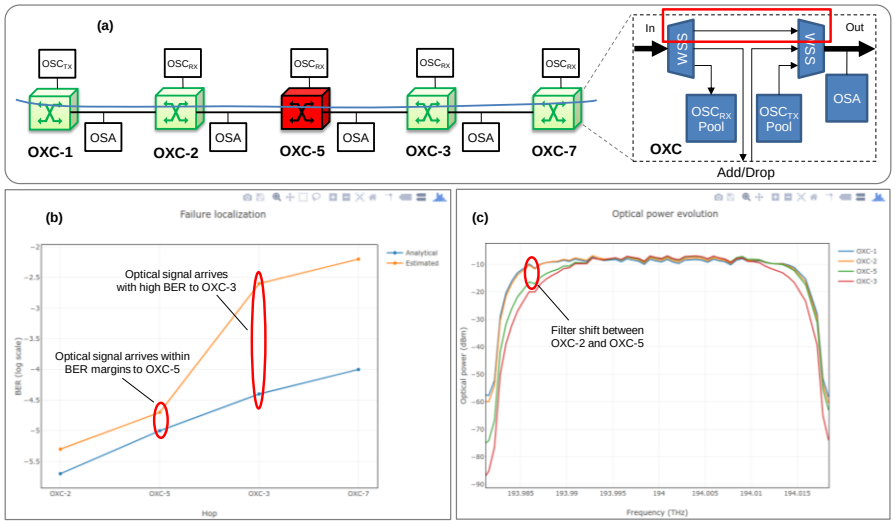

Fig. 4. Illustrative scenario for failure localization (a) using distributed OSCs (b) and OSAs (c).

the devices, different templates are used from the OXC to the hypernode and from that to the data analytics in the C&M plane; specifically, OXC-1 uses template 340 for its OSC_{TX} device, whereas the rest of OXCs use template 350 for their OSC_{RX}. The local KDD processes inside the hypernodes collect OSC_{RX} BER estimations and generate IPFIX messages toward the C&M system (9). The hypernode uses template 351 to send OSC_{RX}

Source	Destination	Protocol	Info
C&M	Hypernode	HTTP	POST /restconf/data/hypernode/nodes[node-id=1]/monitoring/obs-points HTTP/1.1
Hypernode	OXC-1	HTTP2	SETTINGS, HEADERS, DATA, HEADERS
OXC-1	Hypernode	HTTP2	HEADERS, WINDOW_UPDATE, DATA
Hypernode	C&M	HTTP	HTTP/1.1 200 OK (application/yang.data+json)
C&M	Hypernode	HTTP	PATCH /restconf/data/hypernode/nodes[node-id=1]/monitoring/obs-points[obs-domain-id=0][obs-point-id=1]
Hypernode	OXC-1	HTTP2	HEADERS, WINDOW_UPDATE, DATA
OXC-1	Hypernode	HTTP2	HEADERS, DATA, HEADERS
Hypernode	C&M	HTTP	HTTP/1.1 200 OK (application/yang.data+json)
OXC-1	Hypernode	CFLOW	IPFIX flow (64 bytes) Obs-Domain-ID= 0 [Data:340] OSC_{TX}
OXC-2	Hypernode	CFLOW	IPFIX flow (72 bytes) Obs-Domain-ID= 0 [Data:350] OSC_{RX}
OXC-3	Hypernode	CFLOW	IPFIX flow (72 bytes) Obs-Domain-ID= 0 [Data:350] OSC_{RX}
Hypernode	C&M	CFLOW	IPFIX flow (72 bytes) Obs-Domain-ID= 0 [Data:351]
Hypernode	C&M	CFLOW	IPFIX flow (72 bytes) Obs-Domain-ID= 0 [Data:351]

Create OPs (Messages 1-4)
Activate OPs (Messages 5-8)
9
10
Monitoring

Fig. 5. Wireshark capture for the OSC telemetry workflow.

```
'exportTime':'2017-Oct-4 9:15:02',
'observationDomainId': 0,
'dataSets': [{
  'setId': 330,
  'records': [{
    'observationPointId': 1,
    'subTemplateList': {
      'semantics': 'ordered',
      'templateId': 10000,
      'records': [{
        'frequencyTHz': 193.98,
        'rxPowerdBm': -60.01492
      }, ..., {
      }] }   }]  }]
```

Fig. 6. Details of messages (9) from OSAs.

monitoring data to the C&M plane (10), i.e., OSC_{TX} is omitted. Fig. 4b presents the overall graph with BER estimations received from OSC_{RX} and expected BER values computed using analytical models. In view of the different slopes in the segment OXC-5 to OXC-3, an operator can conclude that something happens therein.

The OSA telemetry workflow is a subset of that for the OSCs. The difference relies in the contents of the messages, specifically IPFIX messages for OSA use the *subTemplateList* field defined in [5], which allows to convey a structured list of data records. This is really convenient to encode optical spectrum data, which includes an ordered list of tuples <frequency, power>. Fig. 6 presents an example of the data in one of the IPFIX messages (9) in the OSA telemetry workflow. Likewise in the OSC workflow, local KDD processes in hypernodes controlling OXCs 1, 2, 3, and 5 forward to the C&M system periodic IPFIX messages including OSA measurements for the reference lightpath (since OSAs analyze the whole C-band, KDD processes select just the lightpath's frequency slot). Fig. 4c presents the overall graph plotting the optical spectrum captured at four different OXCs. It is clear, in the view of the figure the signal is asymmetrically filtered before the OSA in OXC-5, thus pointing out a misconfiguration of a WSS in such node (WSSs within the red box in Fig. 4a).

Finally, note that OSAs and OSCs provide complimentary views, and are useful to localize failures more accurately. In this simple example, detecting anomalies in segments OXC-2 to OXC-5 and OXC-5 to OXC-3 through OSAs and OSCs, respectively, leads the operator to guess that there is a problem in OXC-5.

4. Conclusions

An extension to our CASTOR platform has been presented to enable remote configuration of monitoring devices. The architecture has been experimentally demonstrated through a use case facilitating failure location in an optical network using active (OSC) and passive (OSA) monitoring techniques.

References

[1] D. Geisler *et al.*, "Experimental demonstration of flexible bandwidth networking with real-time impairment awareness," *OpEx*, 2011.
[2] L. Velasco *et al.*, "An Architecture to Support Autonomic Slice Networking," *IEEE/OSA JLT*, 2018.
[3] A. P. Vela *et al.*, "Soft Failure Localization during Commissioning Testing and Lightpath Operation," *IEEE/OSA JOCN*, 2018.
[4] M. Dallaglio et al., "Control and Management of Transponders with NETCONF and YANG," *IEEE/OSA JOCN*, 2017.
[5] B. Claise, "Export of Structured Data in IP Flow Information Export (IPFIX)," IETF RFC 6313, 2011.