Building Autonomic Optical Whitebox-Based Networks

Luis Velasco, Andrea Sgambelluri, Ramon Casellas, Lluis Gifre, Jose-Luis Izquierdo-Zaragoza, Francesco Fresi, Francesco Paolucci, Ricardo Martínez, and Emilio Riccardi

***Abstract*—Disaggregation at the optical layer is expected to bring benefits to network operators by enriching the offering of available solutions and enabling the deployment of optical nodes that better fit their needs. In this paper, we assume a partially disaggregated model with transponder nodes for transmission and ROADMs for switching, where each optical node is conceived as a whitebox consisting of a set of optical devices and a local node controller that exposes a single interface to the software-defined networks (SDN) controller. An architecture to provide autonomic networking is proposed, including the SDN controller and supporting monitoring and data analytics capabilities; YANG data models and software interfaces have been specifically tailored to deal with the level of disaggregation desired. To demonstrate the concept of autonomic networking, use cases for provisioning and self-tuning based on the monitoring of optical spectrum have been proposed and experimentally assessed in a distributed test-bed connecting laboratories in Spain and Italy.**

***Index Terms*—Autonomic networking, optical whiteboxes, partially disaggregated networks, software defined networking, YANG data models.**

I. Introduction

SOFTWARE defined networks (SDN) represent one of the most relevant innovations in recent years for the telecom industry; major operators are planning to progressively migrate their networks, including optical transport, to such paradigm aiming at the programmability and automation of connectivity services. However, although some system vendors offer SDN solutions for the optical transport network, these solutions are often dedicated to single-vendor optical domains, managed as single entities, i.e., *fully aggregated*. Although resources are abstracted by the SDN controller at the networking level and exposed via a north-bound interface (NBI) to operations/business support systems (OSS/BSS), internal resource management, monitoring, and control are proprietary solutions, which forces network operators to create multi-vendor networks through control plane interoperability [1]. In addition, many features are not implemented at the SDN controller level, but actually require proprietary Network and Element Managers. This approach, although suitable for large transport networks due to the complexity of managing physical layer impairments in a vendor agnostic way, is not necessarily cost-effective in metropolitan or regional areas, where impairments are less critical because of the reduced distances. With less sophisticated impairement-aware algorithm, there is an opportunity to *disaggregate* individual domains and *converge* into a single, heterogeneous domain consisting of interoperable optical devices from different vendors.

Disaggregation should allow network operators to select and compose individual nodes by selecting the most appropriate vendor solutions for each function, e.g., transponder, fixed or reconfigurable Optical Add-Drop Multiplexer (OADM), line system, control, monitoring, etc. At least two levels of disaggregation can be considered at the optical layer: *i*) *partially disaggregated*, where some degree of disaggregation into optical components takes place, while still having some level of aggregation and abstraction, and *ii*) *fully disaggregated*, where every single optical component exposes its programmability through a control interface. Both models have their pros and cons, e.g., the fully disaggregated model would provide a higher degree of flexibility at the expenses of a higher complexity at the SDN controller, since routing, modulation format and spectrum allocation (RMSA) algorithms [2], [3] would have to deal with larger network topologies.

Optical nodes or even disaggregated subsystems on a blade (e.g., 1-degree OADM consisting on a set of wavelength selective switches (WSS) to be assembled in groups to make a complete multi-degree Reconfigurable OADM -ROADM) are termed optical *whiteboxes*.

Some initiatives (e.g., Open ROADM [4] and OpenConfig [5]) are currently working on defining and implementing multi-source agreements for optical whiteboxes. Open ROADM is focused on a multi-vendor optical network based on ROADM and

Manuscript received November 21, 2017; revised April 14, 2018; accepted April 19, 2018. Date of publication April 24, 2018; date of current version June 15, 2018. This work was supported in part by the EC through the METRO-HAUL (G.A. no. 761727) project, in part by the Spanish MINECO SYNERGY (TEC2014-59995-R) and TWINS (TEC2017-90097-R) projects, and in part by the Catalan Institution for Research and Advanced Studies. *(Corresponding author: Luis Velasco.)*

L. Velasco and J.-L. Izquierdo-Zaragoza are with the Optical Communications Group, Universitat Politecnica de Catalunya, Barcelona 08034, Spain (e-mail: lvelasco@ac.upc.edu; josel.izquierdo@ac.upc.edu).

A. Sgambelluri, F. Fresi, and F. Paolucci are with the Scuola Superiore Sant'Anna, Pisa 56127, Italy (e-mail: a.sgambelluri@sssup.it; francesco.fresi@sssup.it; fr.paolucci@sssup.it).

R. Casellas and R. Martínez are with the Centre Tecnològic de Telecomunicacions de Catalunya, Barcelona 08860, Spain (e-mail: ramon.casellas@cttc.es; ricardo.martinez@cttc.es).

L. Gifre is with the High-Performance Computing and Networking Group, Universidad Autónoma de Madrid, Madrid 28049, Spain (e-mail: lluis.gifre@uam.es).

E. Riccardi is with Telecom Italia, Turin 10148, Italy (e-mail: emilio.riccardi@telecomitalia.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JLT.2018.2829700

flexible transponders as separate optical nodes. Management and physical-level interoperability is specified in detail, including vendor agnostic specifications for commissioning, operation and maintenance procedures. OpenConfig aims to achieve interoperability in the management and monitoring of a multi-level network, realized with different technologies to be able to standardly configure and monitor single networking devices from different vendors.

Even though these initiatives are developing detailed standard south-bound interfaces (SBI), the problem of creating a common platform to build self-managed (*autonomic*) networking [6] is still not completely addressed. Note that autonomic networking entails the capability to do measurements on the data plane, and generating data records that are collected and analyzed to discover patterns (knowledge) from data (KDD). Such knowledge can be used to issue re-configuration/re-optimization recommendations toward the SDN controller. For instance, bit error rate (BER) degradation can be detected on a lightpath and a recommendation to the SDN controller can be issued to reroute such lightpath [7]; this is referred to as a *control loop*. BER, optical power, and other parameters can be metered in optical transponders. In addition, specific monitoring devices can be included to complement the parameters metered by optical devices, e.g., Optical Spectrum Analyzers (OSA) were used in [8] for failure localization purposes.

Thus, to build autonomic networking, a monitoring and data analytics (MDA) system needs to be deployed to complement the SDN controller, the latter focusing on connection provisioning, failure recovery, and network re-optimization (see e.g., [9]). Authors in [10], [11] proposed a distributed MDA architecture, named CASTOR, to enable autonomic networking. CASTOR key features are: *i*) the node extended capabilities to collect monitoring data from the devices and apply algorithms for KDD purposes (referred to as *local KDD*), which enable control loops implementation; and *ii*) a centralized MDA system that collates and analyzes monitoring data records from all the nodes in the network, applies machine learning algorithms and finally, triggers notifications suggesting actions to the SDN controller.

In this paper, we assume the partially disaggregated model and consider two types of optical nodes: *i*) transponder nodes in charge of the optical transmission; and *ii*) ROADMs in charge of optical switching. This model is conceptually similar to the approach considered within Open ROADM and exploits the benefits of disaggregation whilst requiring limited efforts for standardization, compared to the full disaggregated approach. Specifically, the contributions are:

1) Section II discusses the specific requirements to create autonomous networks with optical whiteboxes. From the perspective of the control and management plane, a MDA system will be in charge of implementing control loops, not only at the network level, but also at the node level. In addition, node controllers are needed to coordinate the devices within each node.
2) Section III presents our node proposals for the selected level of disaggregation, as well as the control and management architecture, including MDA, and the specific node and monitoring YANG data models [12].

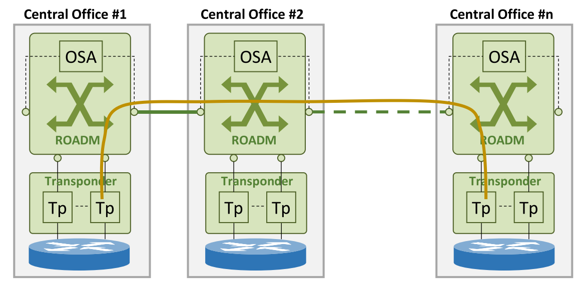


Fig. 1. Example of a whitebox-based network. OSAs are installed in the ROADMs for monitoring purposes.

3) Section IV proposes uses cases targeting at demonstrating connection provisioning, as well as self-tuning on the proposed solution. A specific workflow is defined for each use case, where messages exchanged between control and MDA modules are specified.

The discussion is supported by the experimental demonstration of the use cases presented in Section V, carried out on a distributed test-bed connecting premises in UPC (Barcelona, Spain), CTTC (Castelldefels, Spain), and SSSA (Pisa, Italy).

II. Requirements for Autonomic Optical Whitebox-Based Networking

In this section, we introduce the main concepts and the motivation, as well as some major requirements to create autonomic whitebox-based networks. To take full advantage of disaggregation, a real *ecosystem* should be created and maintained with the involvement of equipment and software vendors, system integrators and standard development organizations to fulfill the requirements of interoperability among optical nodes in a multi-vendor environment, including: *i*) the identification of a common SBI interface for control and telemetry; *ii*) the specification of proper and vendor agnostic commissioning operation and maintenance procedures; *iii*) the development of software products and platforms for SDN control and service, element and network managing; and *iv*) the development of software products for planning and optical design.

As mentioned in the introduction, in this paper we consider a partially disaggregated model with transponder nodes and ROADMs. An example is shown in Fig. 1, where a number of Central Offices (CO) are interconnected; each CO might include several optical nodes. In addition, specific monitoring devices can also be equipped within the optical nodes to complement metering. For instance, the ROADM in CO#2 in Fig. 1 is equipped with OSAs to acquire the optical spectrum at input and output optical interfaces. Note that the available monitoring devices in every node and their monitoring capabilities might be different for every node. As a consequence, it is strictly required to know not only the measurements that can be activated in the nodes, but also the interfaces in which they can be measured.

From the viewpoint of the control plane, an SDN controller is assumed to be in charge of the optical disaggregated network (see Fig. 2). We consider, for our purposes, that the SDN controller maintains several operational databases. In particular, the topology database (TopoDB) stores resource status of the

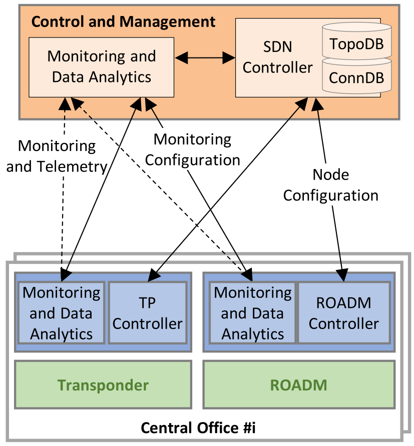


Fig. 2. Overview of the proposed control architecture and control interfaces for optical whitebox-based networks.

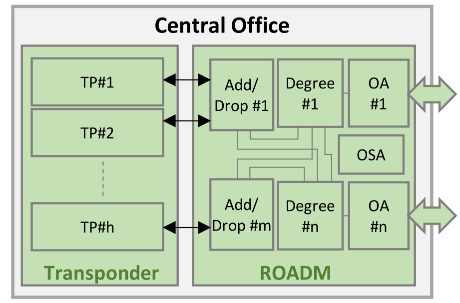


Fig. 3. Optical whiteboxes considered in this paper.

different nodes and inter-connecting links, whereas the connection database (ConnDB) stores the attributes of the connections.

A MDA system is required to enable *autonomic networking*, i.e., to implement control loops that entail monitoring and re-configuration or re-tuning. This system should provide support for *Observation Points* (OPs) management, being an OP, an abstract representation of a given resource in the network where measurements can be obtained. Note that the proposed differentiated roles between the SDN controller and the MDA system allows to take advantage of specialization, as the latter requires from big data capabilities for data collection and analysis.

In addition, two particularities can be highlighted: *i*) a local *node controller* is needed to expose a single interface to the SDN controller and coordinate actions toward the different devices; and *ii*) *control loops* need to be implemented not only at the network level, i.e., spanning different nodes in the network, but also to re-configure or re-tuning different devices within the whitebox. Note that those changes affect either those parameters not being imposed from the SDN controller that give freedom degrees to improve quality, reduce margins, and so on (re-configuration), or parameters that show a deviation with respect to the specific values requested by the SDN controller (re-tuning).

The development of such local node controller is challenging, since it has to deal with the specifics of devices from different vendors, while offering a common, unified view with the right level of abstraction of the underlying hardware. Specifically, a common YANG data model needs to be defined for the control of the optical whiteboxes, and should include configuration-related, as well as monitoring-related parameters.

III. Optical Whitebox-Based Networks

This section presents our solutions for autonomic optical networking under the partially disaggregated model.

A. Control of Optical Whiteboxes

The chosen model identifies two different optical nodes: transponder nodes and ROADMs (see Fig. 3). The internal architecture of the transponder nodes is, for the purpose of this work, straightforward as it just includes a pool of optical transponders (and/or muxponders, switchponders). However, the architecture of ROADMs is much more complex and consists of: *i*) a set of add/drop modules each with a number of interface ports for add and drop client optical signals into the optical network. Add/drop modules might include WSSs and passive mux/demux devices with amplification, if needed; *ii*) a set of nodal degrees modules that include WSSs, variable optical attenuators and optical amplifiers; the number of modules equipped in the ROADM determines its degree; iii) OSAs acquiring optical spectrum at the optical degree interfaces.

Optical nodes need a dedicated node controller responsible for: *i*) abstracting the details of the node towards the SDN network controller, exporting a single unified model regardless of the actual composition or vendor of the node; *ii*) implementing the NETCONF server that allows the network SDN controller (acting as NETCONF client) to configure the node; and *iii*) the subsequent configuration of the individual devices composing the node (such as the add/drop modules or degree components of ROADMs or the individual transponders part of a given transponder pool).

It is worth noting that the control interfaces between the node controller and the individual devices are internal, although nothing precludes that they could be based on YANG/NETCONF, thus having a hierarchy of control and management interfaces and detailed YANG data models.

B. Control Architecture and YANG Data Model

A detailed architecture that includes both, control and MDA, is presented in Fig. 4. We assume that each node controller exports a control and management interface, whose capabilities in terms of configuration, operational status, exported Remote Procedure Calls (RPC), and generated notifications are defined by the YANG data model presented below in this section. NETCONF [13] is a protocol that can be used to access and manipulate the device data model, issue device commands and gather device notifications. This interface is consumed by a logically centralized network-wide SDN controller.

The SDN controller is responsible for service and resource orchestration: starting from a high-level service request coming from OSS/BSS systems via the SDN controller NBI, the SDN controller performs functions such as path computation and resource assignment / allocation, subsequently deriving a sequence of node configuration operations to be committed

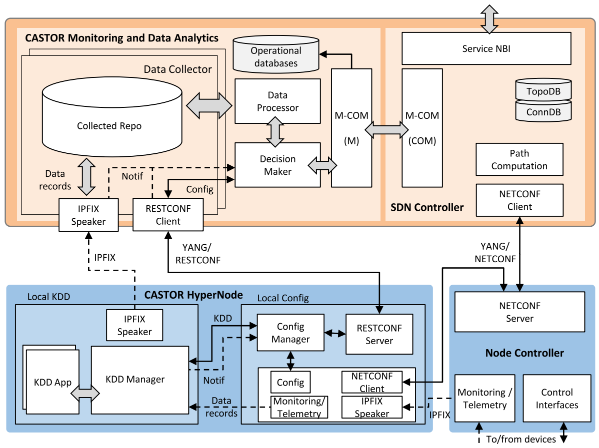


Fig. 4. Detailed control and monitoring architecture for autonomic networking.

using its dedicated SBI towards the node composing the optical infrastructure.

The YANG data model enables: *i*) the fine-tune configuration of the underlying devices; *ii*) the real-time monitoring of key physical parameters and other operational data; *iii)* the implementation of local control loops by automating basic re-configuration/re-tuning in the form of RPCs; and *iv)* a mechanism for event notification.

In order to univocally identify the node, a number of parameters are included in the YANG data model, following a hierarchical tree that includes central office location, node type, the unique node identifier, unique IP address, rack/shelf ID in which the node is installed, and the number of controlled devices under the same controller.

Two differentiated node controller types have been considered, one for transponder nodes (TP) and another for ROADMs. In a TP node controller, a standard and vendor neutral subtree of the YANG model (named *config*) is adopted for each controlled transponder, to store configuration parameters as provided by the SDN controller. For example, the optical interface of a transponder is described as a sequence of attributes (expressed as leaves in the YANG tree) including *central-frequency*, *bit-rate*, *baud-rate*, *slot-width*, *modulation-format, transmit-power*, and *fec*. The operational status of a TP node replicates the same structure. For each transponder, a subtree of the YANG data model (named *state*) is used to keep an up-to-date view of the considered parameters. For example, *input-power*, *pre-fec-ber*, *osnr*, *snr*, *pmd*, *chromatic-dispersion* and *q-factor* are listed as attribute in the YANG tree.

In a ROADM node controller, for each degree module, the config subtree is used to store the running configuration. In particular, the configuration contains a list of established cross-connections with their key attributes including *cross-connection-id*, *input-port*, *output-port*, *central-frequency* and *slot-width*. The operational status of a ROADM is kept up-to-date in the state subtree of the YANG data model, e.g., *input-port*, *output-port* and current *slot-width* values are maintained for each cross-connection in each degree module.

Besides configuration and operational status, specific notifications have been designed to convey specific events. For example, once the SDN controller configures a lightpath using one of the controlled transponders, the TP node controller sends a notification to the registered external entities (see next subsection) for synchronization purposes. The notification encloses the lightpath identifier received from the SDN controller, as well as the local identifier.

C. Autonomic Networking and Interfaces

Regarding the MDA system, although we assume the distributed architecture in [10], a number of extensions have been added to CASTOR to be compliant with the architecture defined in Section II; in particular (see Fig. 4): *i*) the local KDD module has been extended with a local config module (hereafter *hypernode*). Hypernodes allow local control loops implementation, e.g., collecting monitoring data and applying re-configuration/re-tuning to devices in the local node. In this regard, a NETCONF interface based on an extended version of the YANG data model defined in the previous subsection that includes OP management has been added [11]; *ii*) as in [10], the protocol used for monitoring configuration is RESTCONF [14] that it is based on the extended YANG data model; *iii*) a specific interface is required to coordinate the SDN controller and the centralized MDA system. Consequently, an east-west interface, named *M-COM*, has been defined for synchronization and coordination purposes between the MDA system (M) and other Control, Orchestration, and/or Management modules (COM), e.g., the SDN controller; and *iv*) the IPFIX protocol [15], used for monitoring purposes, has been extended as follows: apart from the template to convey BER and optical power metered in the transponders defined in [10] (*templateId 310*), we have defined another one to convey optical spectrum measurements (*templateId 330*) that basically contains a list of tuples <*frequency*, *power*> encoding a subTemplateList field [15]. The contents of monitoring data records generated during the metering process are defined by specifying the monitoring *templateId*.

As mentioned above, the YANG data model defined in Section III-B has been extended to include monitoring configuration; a new subtree, named *monitoring*, is included. The monitoring subtree enables not only auto-discovery of monitoring capabilities but also configuration of OPs. The intention is that the MDA system to be able to correlate information in operational databases (i.e., nodes, links, connections, etc.) with the monitoring capabilities retrieved from the node controllers. In addition, the YANG data model provides support for asynchronous notifications issued by hypernodes toward the MDA system, as well as the support for RPCs to be called from the MDA system and executed in the local KDD of a hypernode, aiming at implementing control loops that entail remote device re-tuning.

Let us now specify the functional requirements of the M-COM interface. In dynamic scenarios, resources are created, modified, and removed. In consequence, the M-COM interface needs to support not only retrieving and synchronizing the contents of the aforementioned operational databases from COM modules (which are needed to correlate measurements to resources), but also include *notifications* on resource changes.

In addition, the M-COM interface needs to support event notifications that might include recommended actions. For example, *Network Event*s can be defined as the result of detecting specific conditions or patterns after monitoring data collated from OPs are analyzed, and recommendations may involve network reconfiguration. Specifically, six primitives are considered, where the notation X→Y specifies that X is the primitive requestor:

- *Get list of databases* (M→COM): returns the list of databases in the COM module. Each database is defined by the tuple <*Database-Identifier*, *Database-Name*, *Resource-Type list*>, where the defined resource types include node, link, and connection.
- *Get database* (M→COM): returns the contents of the specified database. Each resource is represented by a JSON object that contains the attributes of the resource.
- *Subscribe/unsubscribe to database changes* (M→COM): subscribes/unsubscribes to asynchronous notifications on changes in the selected database.
- *Notify database change* (COM→M): notifies about changes in resources within the specified database. The new JSON objects representing the changed resources are included.
- *Subscribe/unsubscribe to network events* (COM→M): subscribes/unsubscribes to asynchronous events issued during monitoring and data analytics processes.
- *Notify network event* (M→COM): notifies about specific conditions or patterns found in the network during monitoring and data analytics processes. The notification can include a recommendation to the COM module.

The next section defines a set of use cases that exploit the architecture and functionalities described in this section to create autonomic optical networks.

IV. Illustrative Use Cases

In this section we propose use cases aiming at illustrating provisioning and control loops on a network with optical whitebox nodes. Let us assume a lightpath between COs#1 and #3 traversing CO#2, where the ROADM node in such CO#2 is equipped with an OSA that acquires the spectra in the input and the output optical links. Let us imagine that two different soft failures, i.e., events that progressively degrade the quality of the signal, affect the lightpath (see [7] and [8]): *i*) a misconfiguration that result in a filter shift or tight filtering failure. Such misconfiguration can be discovered locally by detecting the failure in the spectrum acquired in the output interface and by confirming that the same failure is not affecting the signal at the input interface, and it might be corrected by re-tuning the local WSSs; *ii*) a laser drift problem that can be detected by analyzing the optical spectra (the central frequency of the signal is deviated from its nominal value), but re-tuning has to be applied in a remote node.

The use cases are defined by the following workflows: *i*) lightpath provisioning (WF1); *ii*) local control loop after the detection of a filter shift failure in the local ROADM (WF2); and *iii*) network control loop after the detection of a laser drift in the remote Transponder node (WF3). Fig. 5 shows the message exchanges corresponding to each workflow.

Before workflows execution, all the modules are synchronized. Specifically, the centralized MDA system is assumed to already have retrieved operational data from the network controller using the M-COM *Get database* primitive. In addition, subscription to database changes and event notifications have been requested.

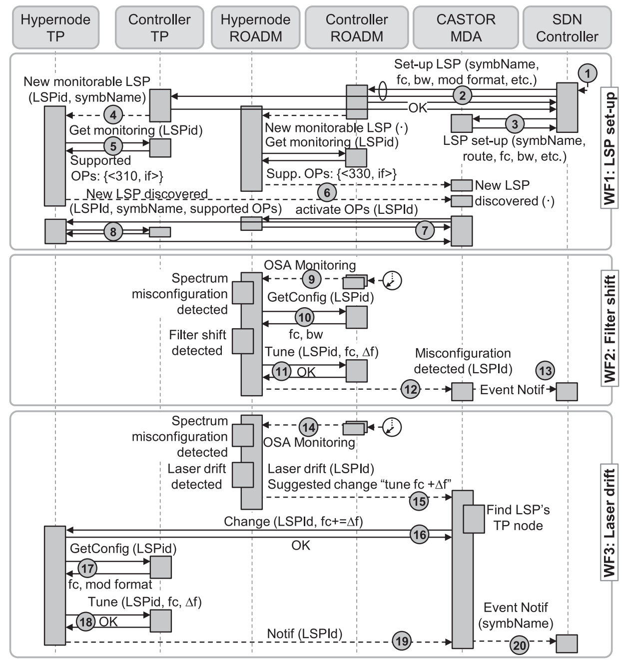


Fig. 5. Provisioning (WF1) and control loops (WF2 and 3) for self-tuning workflows demonstrated in this paper.

WF1 is triggered by the reception of a connection set-up request in the SDN controller (message 1 in Fig. 5). After solving the RMSA problem, the SDN controller issues control messages through the NETCONF interface to every node in the route of the lightpath specifying the central frequency (*fc*), spectrum bandwidth (*bw*), modulation format, etc., as well as the symbolic name of the lightpath (messages 2). When confirmation is received from all the nodes, the lightpath state is changed in ConnDB and a notification (*Notify database change*) is sent to the MDA system with the details of the established lightpath (message 3). Note that no monitoring capabilities on the lightpath are available until they are discovered at the node level. To that end, node controllers send an asynchronous notification to their peer hypernodes as soon as the resources for the new lightpath have been configured in the internal devices in the node and monitoring capabilities are confirmed (message 4); the message specifies among others, the symbolic name of the lightpath received from the SDN controller, as well as the local identifier (*LSPid*). The hypernode can now request the available OPs for the lightpath that include both, the monitoring method identified by the *templateId* and the interface where measurements will be performed. To this end, the hypernode issues a *get* message specifying the received *LSPid* (message 5). Finally, hypernodes send a notification toward the MDA system reporting the discovered new lightpath together with the available OPs (messages 6).

Once the lightpath is in operation, OPs can be activated. For instance, BER and optical power can be measured by activating *templateId* 310 in the transponders, whereas optical spectrum can be acquired in the input and output interfaces of the ROADM by activating *templateId* 330 for such interfaces (messages 7 and 8). In the case of optical spectrum, we assume that the whole C-band is being already acquired as a result of active OPs for the optical links and message 8 is not issued to the ROADM node controller; instead, a process inside the hypernode is in charge of selecting the portion of the spectrum for the LSP. After such activation, monitoring data records with measurements are issued periodically by the node controllers towards the hypernodes and analyzed by specific KDD applications in the hypernodes. In the case of OSAs, the corresponding KDD process (named Signal Spectrum Verification- *SSV*) in the hypernode checks the metered spectrum to detect soft failures affecting the lightpath.

The second workflow (WF2) assumes a filter shift failure originated by a misconfiguration problem in a WSS in the local ROADM. The acquired spectra in the input and the output links are received periodically by the hypernode in CO#2 from the ROADM controller (message 9). The SSV algorithm first extracts the set of features that characterize the optical spectrum; features include the measured central frequency and bandwidth at different power levels. A machine learning algorithm analyzes those features and detects whether the sample is normal or is affected by a soft failure (see details in [8]). In the case that a failure is detected after analyzing the spectrum in the output link, that in the input is analyzed to determine whether the failure is localized in the current node. In such case, the current configuration of the nodal degree modules that take part in the lightpath switching is requested to the ROADM controller (message 10). In case that either the configured central frequency or the filters bandwidth do not match with the metered ones, a filter problem (e.g., filter shift) has been confirmed and a re-tuning of the local devices inside nodal degrees is requested by issuing a message to the node controller (message 11). Finally, the hypernode issues a notification to the MDA system (message 12) that generates an event notification toward the SDN controller (message 13).

The last workflow (WF3) assumes a laser drift failure originated by a problem in CO#1. As in WF2, the acquired spectrum is received periodically by the hypernode in CO#2 from the ROADM controller (message 14) and the SSV algorithm in the hypernode detects a spectrum misconfiguration. After confirming that the problem is not in the local node and evaluating the magnitude of the laser drift failure, a notification is issued toward the MDA system suggesting re-tuning the central frequency of the laser in the transponder (message 15). Upon the reception of the notification, the MDA system finds the Transponder node terminating the lightpath (note that such data was synchronized during WF1 from the SDN controller) and issues a request toward the hypernode in CO#1 (message 16). The hypernode first confirms the laser drift problem by getting the current configuration of the transponder to the node controller (message 17) and then requests a laser tuning of the magnitude of the failure detected (message 18). Finally, the hypernode replies to the MDA system (message 19) confirming the laser re-tuning operation, and an event notification is generated toward the SDN controller (message 20).

The next section demonstrates the proposed use cases on our experimental test-bed.

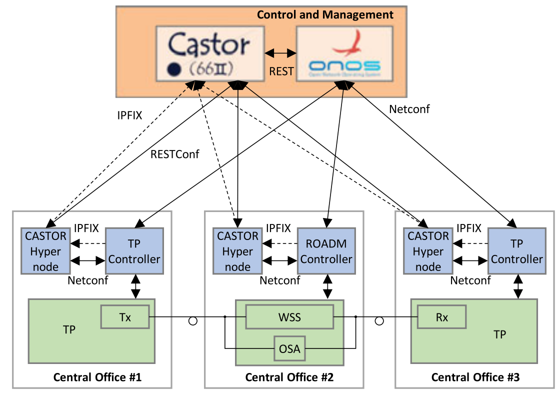


Fig. 6. Experimental test-bed.

V. Experimental Assessment

A. Test-Bed Description

The experimental assessment has been carried out in a distributed test-bed connecting UPC (Barcelona, Spain), CTTC (Castelldefels, Spain), and SSSA (Pisa, Italy) premises. The optical network and the control and management architecture depicted in Fig. 6 were deployed, where the modules running in different locations where connected through IPSec tunnels.

The optical infrastructure is in SSSA premises; in particular, two transponder nodes are deployed in CO#1 and #3, which consist of Ericsson PM-QPSK 100G Metro 10x-muxponder cards, based on the SPO-1400 ROADM platform (*vendor A*). Moreover, one ROADM deployed in CO#2 consists of one Finisar WSS and one OSA (*vendor B*). Transponders and ROADM nodes are connected by means of 2x 160 km-long SMF-based optical links with independent EDFA amplification stages. Node controllers supporting the YANG models defined in Section III are implemented in Python/C++ and are based on the *ConfD* software tool [16], providing a NETCONF interface to the SDN Controller for configuration purposes (i.e., edit-config commands) and toward the local hypernode for monitoring purposes (i.e., notification subscription and get-config commands). In addition, a Python-based IPFIX module is responsible for exporting monitoring data records, including OSA spectrum samples. In particular, the OSA is configured to sample the C-band optical spectrum alternately at the input and at the output port of the WSS.

CTTC's SDN Controller is implemented extending the ONOS controller framework [9]. A service YANG model is registered in the ONOS YANG subsystem, so a RESTCONF based interface can automatically be used to trigger service provisioning. Regarding device and network topology management, a dedicated REST NBI is also used to provision the controller with the IP addresses, ports and required credentials for the 3 NETCONF devices corresponding to the 3 optical nodes, along with their interconnection (the network topology is thus not discovered but managed). From the SBI point of view, new ONOS *behaviors* have been defined (that is, abstractions of various device configuration or device adaptations), covering a transponder and a ROADM node. Consequently, new *drivers* implementing such behaviors have been added, hiding driver-specific details while mapping high-level operations to NETCONF operations. For

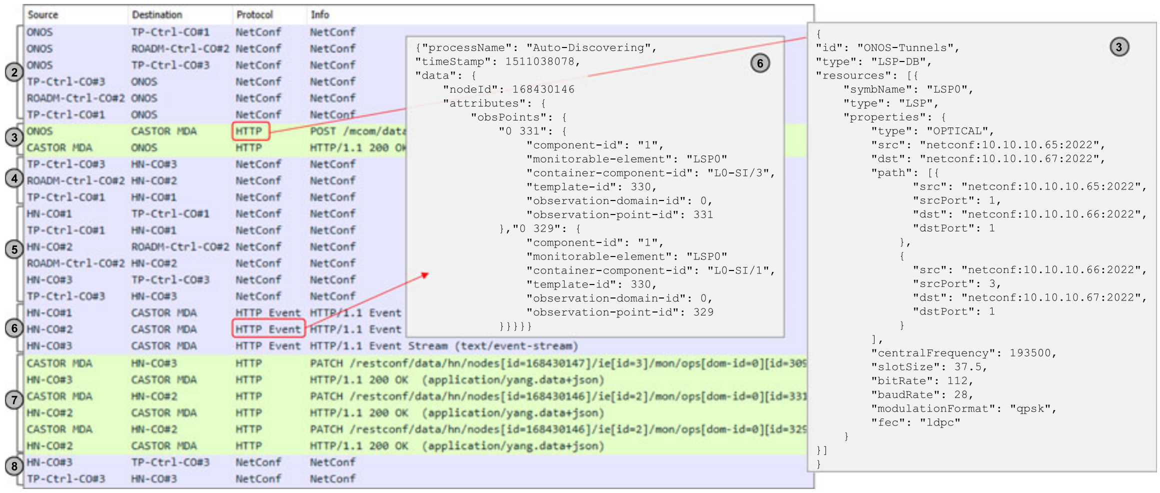


Fig. 7. Message list for connection provisioning (WF1) and details of messages 3 and 6.

	Source	Destination	Protocol	Info
9	ROADM-Ctrl-CO#2	HN-CO#2	CFLOW	IPFIX flow (1236 bytes) Obs-Domain-ID= 0 [Data:330]
	ROADM-Ctrl-CO#2	HN-CO#2	CFLOW	IPFIX flow (1236 bytes) Obs-Domain-ID= 0 [Data:330]
10	HN-CO#2	ROADM-Ctrl-CO#2	NetConf	NetConf
	ROADM-Ctrl-CO#2	HN-CO#2	NetConf	NetConf
11	HN-CO#2	ROADM-Ctrl-CO#2	NetConf	NetConf
	ROADM-Ctrl-CO#2	HN-CO#2	NetConf	NetConf
12	HN-CO#2	CASTOR MDA	HTTP Event Stream	HTTP/1.1 Event Stream (text/event-stream)
13	CASTOR MDA	ONOS	HTTP Event Stream	HTTP/1.1 Event Stream (text/event-stream)

Fig. 8. Message list for local self-tuning (WF2).

example, the *TransponderConfigBehaviour* abstracts transponder parameter configuration, or the *CrossConnectBehavior* abstracts the cross-connection of a flexi-grid frequency slot from one node input port to a node output port.

Finally, UPC's CASTOR MDA and hypernodes are developed in Python and run in a computer cluster under Linux. CASTOR MDA is connected to the ONOS instance through the M-COM interface described in Section III-C, implemented as REST interface(s), where both CASTOR MDA and ONOS act as client/servers depending on the actual function. CASTOR hypernodes use a NETCONF interface based on the YANG data model described in Section III to connect to node controllers, whereas an IPFIX interface is used for monitoring purposes.

B. Experiments

Starting with WF1, Fig. 7 presents the list of exchanged messages. After receiving a request through the service NBI, the SDN controller issues messages to every node controller in the route of the lightpath (messages 2) specifying the parameters for the transponders for TP node controllers in CO#1 and #3 and those for the cross-connection for ROADM controller in CO#2. Once confirmations arrived, the SDN controller notifies the new lightpath to the MDA system through the M-COM interface. The details of message 3 shown in Fig. 7 bring together the contents of messages 2 that the SDN controller sent to the individual node controllers. In particular, the connection's *symbName* ("LSP0") is specified together with its route and the *fc* and *bw* of the signal. Simultaneously, the node controllers have notified their peer hypernodes about resources have been allocated for a new connection (messages 4); hypernodes obtain the OPs


```
(11)
<config><roadm-degree-ne xmlns:nc=
      "urn:ietf:params:xml:ns:netconf:base:1.0"
      xmlns="http://sssup.it/filter">
    <degree-lsp-attributes>
        <degree-id>1</degree-id>
        <lsp-id>1</lsp-id>
        <input-port>1</input-port>
        <output-port>3</output-port>
        <nominal-central-freq>193500</nominal-central-freq>
        <slot-width>37.5</slot-width>
        </degree-lsp-attributes>
</roadm-degree-ne></config>
```


Fig. 9. Details of message 11 for filter shift re-tuning.

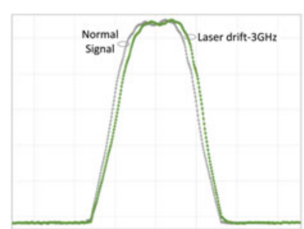
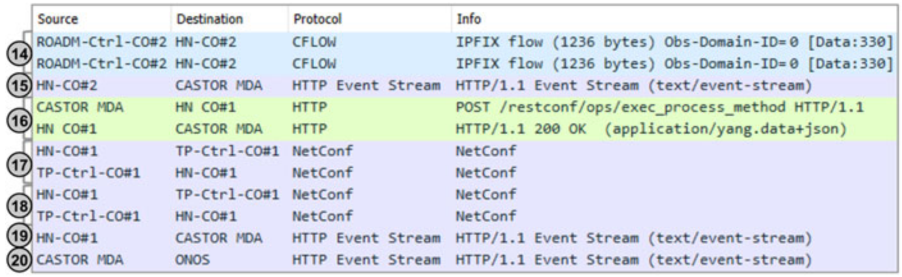


Fig. 10. Original and tuned optical signals after laser drift.

that can be activated for the lightpath (messages 5) and trigger a notification to the MDA system with the discovered information (messages 6). For instance, the hypernode in CO#2 includes two OPs available for the connection (identified by both, its local id and by the symbName given by the SDN controller), one in port 3 and another in port 1; *templateId* 330 (OSA) will be used to the measurements in both OPs. Upon the reception of the available OPs, the MDA system decides to activate all three (one OP with *templateId* 310 in the Rx transponder in CO#3 and two OPs in the ROADM in CO#3). Finally, OP activation is sent to the TP controller in CO#3; recall that OSA monitoring was already active for both links.



	Source	Destination	Protocol	Info
14	ROADM-Ctrl-CO#2	HN-CO#2	CFLOW	IPFIX flow (1236 bytes) Obs-Domain-ID= 0 [Data:330]
	ROADM-Ctrl-CO#2	HN-CO#2	CFLOW	IPFIX flow (1236 bytes) Obs-Domain-ID= 0 [Data:330]
15	HN-CO#2	CASTOR MDA	HTTP Event Stream	HTTP/1.1 Event Stream (text/event-stream)
16	CASTOR MDA	HN CO#1	HTTP	POST /restconf/ops/exec_process_method HTTP/1.1
	HN CO#1	CASTOR MDA	HTTP	HTTP/1.1 200 OK (application/yang.data+json)
17	HN-CO#1	TP-Ctrl-CO#1	NetConf	NetConf
	TP-Ctrl-CO#1	HN-CO#1	NetConf	NetConf
18	HN-CO#1	TP-Ctrl-CO#1	NetConf	NetConf
	TP-Ctrl-CO#1	HN-CO#1	NetConf	NetConf
19	HN-CO#1	CASTOR MDA	HTTP Event Stream	HTTP/1.1 Event Stream (text/event-stream)
20	CASTOR MDA	ONOS	HTTP Event Stream	HTTP/1.1 Event Stream (text/event-stream)

Fig. 11. Message list for remote self-tuning (WF3).

Fig. 8 lists the exchanged messages for WF2. After receiving the measured spectrum for ports 1 and 3 in ROADM node in CO#2 (messages 9), the SSV KDD process detects a filter shift in the local node, so it first retrieves the current configuration of all the WSSs that support LSP0 (messages 10) and detects a misconfiguration in the filters responsible for the filter shift problem. In such scenario, the hypernode decides to re-tune the filters to the right *fc* (messages 11); see the details of the XML-formatted contents in Fig. 9. Note that this action corrects the filter misconfiguration problem by enforcing the parameters initially configured by the SDN controller. The re-tuning action is notified to the MDA system (message 12), which forwards it to the SDN controller for information purposes (message 13).

Finally, the exchanged messages for WF3 are listed in Fig. 11. The list is similar to that for WF2 with the exception of tuning is performed on the central frequency of the transmitter laser in a different optical node; the hypernode in CO#2 detects the laser drift failure and sends a notification to a process running in the MDA system (message 15), which executes a RPC in a process running in the hypernode in CO#1 (messages 16), which gets the current configuration from the local TP node controller (messages 17) and requests laser retuning (messages 18). Fig. 10 presents the acquired spectrum with the OSA when the laser drift was detected and after laser re-tuning.

VI. Concluding Remarks

An architecture for autonomic optical networking has been proposed assuming the partially disaggregated model, where two types of optical whiteboxes were considered for transmission and switching, respectively. The architecture of Transponder nodes consists of a set of transponders, whereas that of the ROADMs is more complex as it consists of a set of interconnected add/drop modules, nodal degrees, optical amplifiers, as well as monitoring devices.

To control the optical whiteboxes, a node controller that exposes a single NETCONF interface to the SDN controller was defined. The interface is based on a generic YANG data model for configuration and operational status purposes that hides the internal complexity of the whitebox. A requirement to build autonomic networks is the capability to perform measurements; therefore, an interface to export monitoring data records from measurements performed by the internal devices in the node is also available in node controllers.

A MDA architecture was adopted and extended from [10]. Although hypernodes use the NETCONF interface exposed by the node controller, the YANG model was extended to add monitoring configuration. The centralized MDA system is collocated with the SDN controller and requires some sort of synchronization regarding the state of the network. To that end, the M-COM interface was specified.

The architecture was demonstrated by defining use cases for provisioning and local and remote self-tuning control loops. The use cases were experimentally assessed in a distributed test-bed connecting premises in Spain and Italy.

References

[1] O. González de Dios *et al.*, “Experimental demonstration of multi-vendor and multi-domain elastic optical network with data and control interoperability over a Pan-European test-bed,” *J. Lightw. Technol.*, vol. 34, no. 7, pp. 1610–1617, Apr. 2016.

[2] L. Velasco, A. P. Vela, F. Morales, and M. Ruiz, “Designing, operating and re-optimizing elastic optical networks,” *J. Lightw. Technol.*, vol. 35, no. 3, pp. 513–526, Feb. 2017.

[3] L. Velasco and M. Ruiz, *Provisioning, Recovery and In-Operation Planning in Elastic Optical Networks*. Hoboken, NJ, USA: Wiley, 2017, ISBN 978-1-119-33856-7.

[4] Open ROADM. [Online]. Available: http://www.openroadm.org.

[5] OpenConfig. [Online]. Available: http://openconfig.net/.

[6] M. Behringer *et al.*, “Autonomic networking: Definitions and design goals,” IETF IRTF 7575, 2015.

[7] A. P. Vela *et al.*, “BER degradation detection and failure identification in elastic optical networks,” *J. Lightw. Technol.*, vol. 35, no. 21, pp. 4595–4604, Nov. 2017.

[8] A. P. Vela *et al.*, “Soft failure localization during commissioning testing and lightpath operation [invited],” *J. Opt. Commun. Netw.*, vol. 10, no. 1, pp. A27–A36, Jan. 2018.

[9] Open Network Operating System (ONOS). [Online]. Available: https://onosproject.org/.

[10] L. Velasco *et al.*, “An architecture to support autonomic slice networking [invited],” *J. Lightw. Technol.*, vol. 36, no. 1, pp. 135–141, Jan. 2018.

[11] L. Gifre, J.-L. Izquierdo-Zaragoza, M. Ruiz, and L. Velasco, “Autonomic disaggregated multilayer networking,” *J. Opt. Commun. Netw.*, vol. 10, no. 5, pp. 482–492, 2018.

[12] M. Bjorklund, “YANG - A data modeling language for the network configuration protocol (NETCONF),” IETF RFC 6020, 2010.

[13] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, “Network configuration protocol (NETCONF),” IETF RFC 6241, 2011.

[14] A. Bierman, M. Bjorklund, and K. Watsen, “RESTCONF Protocol,” IETF RFC 8040, 2017.

[15] B. Claise, G. Dhandapani, P. Aitken, and S. Yates, “Export of structured data in IP flow information export (IPFIX),” IETF RFC 6313, 2011.

[16] ConfD. [Online]. Available: http://www.tail-f.com/management-agent/

Authors’ biographies not available at the time of publication.