

Control of Open and Disaggregated Transport Networks using Open Network Operating System (ONOS) [Invited]

ALESSIO GIORGETTI^{1,2,*}, ANDREA SGAMBELLURI¹, RAMON CASELLAS³, ROBERTO MORRO⁴, ANDREA CAMPANELLA⁵, AND PIERO CASTOLDI¹

¹Scuola Superiore Sant'Anna, Pisa Italy

²CNIT, Pisa, Italy

³CTTC/CERCA, Barcelona, Spain

⁴TIM, Torino, Italy

⁵Open Networking Foundation, Menlo Park, US

* Corresponding author: alessio.giorgetti@santannapisa.it

Compiled October 4, 2019

Use of disaggregated equipment in optical transport networks is emerging as an attractive solution to bring flexibility and break vendor lock-in dependencies.

The disaggregation process requires standard protocols and interfaces between the control plane and network equipment. NETCONF has been selected as the standard protocol and multiple initiatives are currently working on the definition of standard models for each type of data plane devices. Different levels of disaggregation of the data plane are under evaluation, and it is still not clear up to which level it will be useful to disaggregate the data plane.

The disaggregation of optical networks yielded the development of several SDN-based controllers providing an environment for creating and deploying networking application on optical networks. Among them, the ONOS controller features the most active community with the recent establishment of the ODTN working group, specifically focused on the introduction of required functionality to control and monitor disaggregated transport networks.

This paper reports on the state-of-art, potentials and limitations of the ONOS controller applied to disaggregated optical networks with specific focus on the on-going activities within the ODTN working group. Then, the paper describes a set of experiments performed on a setup including both emulated and real optical devices controlled with ONOS. The performed experiments consider both the establishment of a connectivity service and the recovery of the connectivity in case of failure on the data plane.

© 2019 Optical Society of America

<http://dx.doi.org/10.1364/jocn.XX.XXXXXX>

1. INTRODUCTION

The persistent internet traffic growth, mainly driven by the pervasive deployment of cloud and video services, is asking continuous investments on the telecommunication infrastructure. This process deeply involves the optical transport plane, for which, new control and managements paradigms are emerging mainly based on the Software Defined Networking (SDN) architecture that improves network flexibility and manageability [1, 2].

Optical communication systems based on SDN control are already available on the market and in phase of deployment on the field. However, they typically implement fully aggregated systems providing both transmission and switching functionality with proprietary software for control and configuration. The current challenge is therefore twofold: vertically, as in the most traditional SDN approach, to separate the control plane from the data plane opening the system to **third-party** controllers; horizontally aiming at decomposing the optical communication

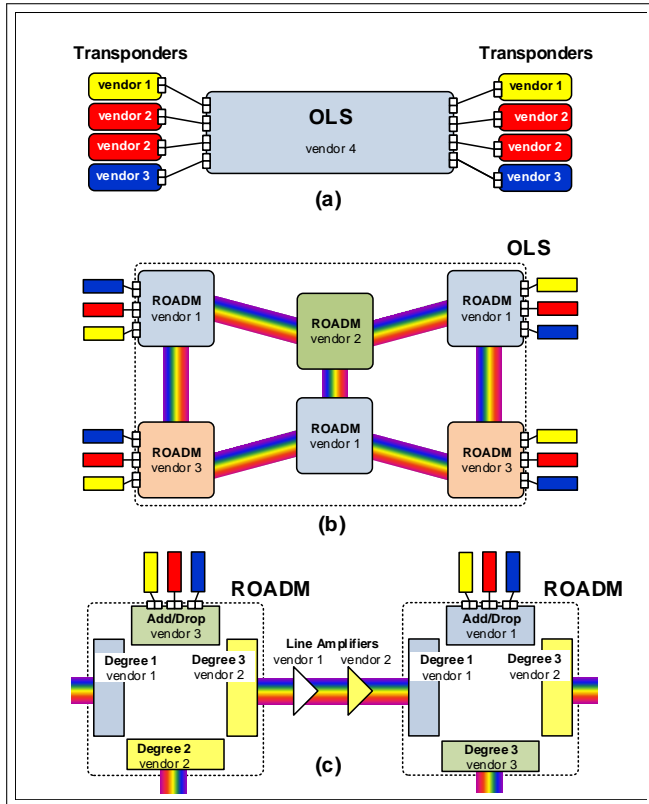


Fig. 1. Disaggregation levels.

system in its single components, allowing best of breed selection during the deployment of optical networks, from a multi-vendor pool, breaking the vendor lock-in dependencies and achieving significant cost reduction [3, 4].

The first step for enabling vertical disaggregation is the standardization of well-defined interfaces between the SDN controller and the data plane to bypass proprietary control and management systems. This includes the choice of a reference communication protocol and the definition of a specific abstraction for each type of data plane devices. The same process took place in the packet switching world when SDN was first introduced through the OpenFlow specification [5] that accomplished both targets: defining the OpenFlow protocol and the abstraction of the switch based on the concept of a pipeline of match and action tables.

On the one hand, regarding the communication protocol, a first attempt to enable SDN in optical networks has been made through extension of the OpenFlow protocol [6, 7]. However, the process needed to define the required extensions, to pass-through the standardization procedures, and to deploy the extensions on real devices has been demonstrated to be not dynamic enough to be adapted to optical communications where technological innovations are frequently appearing, potentially requiring continuous modification to the protocol. Therefore, the community converged on the utilization of an already well-defined protocol, i.e., NETCONF, that has the important strength to use an Extensible Markup Language (XML)-based data encoding thus not requiring extension at the protocol level for transporting optical configuration data. On the other hand, work is needed to define a standard abstraction of data plane devices [8]. Specifically, relevant work is on-going in the definition of

multi-source agreements and YANG models (e.g., OpenConfig, OpenROADM, T-API and Telecom Infra Project initiatives [9–12]). This way, an SDN controller can build on standard YANG models and procedures to consistently control, configure and monitor the optical network. In this context, the most important network vendors are already selling their product including in the documentation the adopted (not standard) YANG models. Such documentation enables the control of the device using **third-party** controllers, also if the development of a specific drivers **would be still required for each new device to be supported** [13]. Thus, proprietary YANG models do not allow plug and play but enable operators and service providers to start the innovation of the offered services through the implementation of specific applications on the top of the controller. **While waiting commercial products supporting standard YANG models, the research community started the development of NETCONF-based software agents (e.g., METRO-HAUL project [14, 15]) implementing those models to act as a translation point between the SDN controller and commercial or experimental optical devices, thus enabling accurate testing of controller features and performance.**

The horizontal disaggregation of the data plane is an ongoing process that is typically featured in three levels as depicted in Fig. 1 [4]. Specifically, at a first level of disaggregation the data plane is presented as a set of terminal devices (i.e., transponders) and Optical Line Systems (OLS), see Fig. 1(a). With this solution, the operators can independently select and upgrade transponders, which typically have a shorter technological life-cycle with respect to line-systems. Fig. 1(b) shows the second level of disaggregation where the data plane is seen as a number of transponders and reconfigurable optical add-drop multiplexers (ROADMs): in this scenario the SDN controller is enabled to deploy third-part applications for using advanced traffic engineering solutions, i.e., routing and spectrum assignment algorithms (RSAs), that are still not feasible disaggregating only transponders from line-systems. Finally, going deeper in the disaggregation process, each ROADM can be further decomposed in a set of ROADM degrees or even more elementary components (e.g., optical filters and optical amplifiers), as illustrated in Fig. 1(c). As an example, there are already commercially available products disaggregating the optical nodes into single node degrees [2, 13].

From the control plane point of view, besides proprietary controllers typically owned by device vendors, several open-source initiatives recently emerged for the control of disaggregated optical networks, mainly arising from the traditional SDN community (e.g., OpenDaylight [16], ONOS [17]). Among them, the ONOS controller (i.e., Open Network Operating System) has been proposed by the Open Networking Foundation (ONF) with the support of many of the most important telecommunication vendors and operators. Specifically, ONOS is part of a wider set of initiatives promoted by the ONF, that includes other related projects such as the development of a layer-2 network emulator (i.e., MININET), the development of a resource orchestrator (i.e., XOS), the modeling of standard interfaces toward transport networks (i.e., Transport API [18]) and many network applications. Such applications are devoted to specific use cases such as CORD (i.e., **Central Office Re-architected as a Datacenter**), for the utilization of SDN in access and metro networks [19], and the recently created Open and Disaggregated Transport Network (ODTN) project targeting the extension of ONOS to control and monitor disaggregated optical networks.

This paper builds on the work presented in [20]. Specifically,

this work reports a more detailed overview of the state of art on the SDN control of disaggregated optical networks, then the report of the ODTN development activity at ONF has been updated including the work that took place in the last months. Finally, while [20] only reported preliminary results on the connectivity setup obtained in a testbed not including real devices, this work presents a detailed experimental demonstration that uses the ONOS controller to control several real optical devices and evaluates the achievable performance not only in the setup of a connectivity service, but also for recovering the traffic in case of failures on the data plane.

2. PREVIOUS WORK

This section reports recent and on-going activities for enabling the SDN control of disaggregated optical networks. For less recent work focused on the application of SDN concepts to transport networks the readers can refer to [21].

The work reported in [2] investigates a partially disaggregated optical network architecture where multi-vendor transponders can be deployed using a single OLS, thus considering the first level of disaggregation as depicted in Fig. 1(a). It assumes that the utilization of inter-operable transponders may bring to sub-optimal optical transmission performance, and also under this assumption it demonstrates that disaggregated solutions have the potential to reduce the optical-electrical-optical interfaces count, thus resulting in potential cost reduction.

The works in [22–24] are focused on the introduction of monitoring/telemetry functionality in disaggregated optical networks with the aim to introduce autonomicity in the network to further simplify the network management and reduce the operational costs. With this aim those works propose the implementation of a monitoring and data analytic (MDA) tool to inter-operate with the controller, to collect telemetry information and consequently trigger the reconfiguration of the network. The three works operates at different levels, while [22] is focused on definition of the MDA architecture and proper YANG models, [23] proposes a first way in which the MDA tool can interact with the SDN controller to activate monitoring flows. Finally, [24] explores the possibility to utilize the gRPC protocol to provide a number of parallel telemetry streams to the MDA system, without overloading the SDN controller.

In [25] the authors concentrate on how the flexibility introduced by the disaggregation of the network can be exploited for enabling gradual upgrading of nodes components (i.e., node degrees) and for reducing cost and times required to recover the network after possible disasters.

The vision of the TIP consortium is reported in [12], that in an experimental environment deploying multi-vendor transponders on a common line-system is developing an open source tool for quality-of-transmission (QoT) assessment based on the GN model for nonlinear fiber propagation [26]. That kind of QoT evaluation tools should then operate in coordination with the SDN controller for applying advanced RSA algorithms on the network. Also [27] is focused on the TIP vision, describing the software architecture adopted for the operating system of the transponders (e.g., Cassini transponder) implementing the OpenConfig standard model.

The work in [28] reports on the utilization of external planning tools for resource assignment optimization in fully disaggregated optical networks. The planning tool described in [29] is integrated at the orchestrator level, and the obtained results show that the disaggregation can lead to better resource uti-

lization introducing a negligible complexity increase during network discovery phase.

Regarding the utilization of ROADM whiteboxes in SDN-based optical networks, [30] describes the open design of a ROADM that is commercially available, and part of the TIP consortium. It also evaluates the performance of the ROADM in terms of switching time, power consumption, and degradation introduced on the traversing optical signal. Also, it describes the modeling of the ROADM based on the models defined in [13] and the implementation of an interface toward a generic SDN controller using the NETCONF protocol. Other commercially available ROADMs and transponders have been included in experimental demonstrations deploying SDN control using the ONOS controller [31].

The works presented in [32, 33] are based on extended versions of OpenDayLight [16] for the control of optical networks. Specifically, [32] reports on the development activity that is ongoing in the OpenDayLight community for the implementation of a Transport PCE inside the controller. Some preliminary experimental results, obtained in a multi-vendor testbed including real optical devices, are reported demonstrating setup times for an optical connectivity service of few minutes. The authors of [33] extended the OpenDayLight functionality for enabling the setup of optical connectivity services in space division multiplexing (SDM) optical networks. They provide experimental results obtained on an emulated testbed (i.e., not including real optical devices) demonstrating setup time of about one second.

3. ONOS POTENTIALS, PERFORMANCE AND LIMITATIONS

ONOS has been originally designed and implemented to operate on packet switching SDN networks, mainly utilizing the OpenFlow protocol. However, with respect to other SDN controllers, ONOS natively addressed the most common weaknesses of the SDN architecture, i.e., reliability and scalability. Therefore, ONOS is currently one of the best candidates for the control of optical networks, where reliability and scalability are key focus points [34].

From the reliability point of view, ONOS is designed to run in a logically centralized but physically distributed fashion, where the controller functionality is spread over a number of synchronized instances running on different physical machines [35]. Since version 1.4 of ONOS the Atomix framework is used, where the databases for devices, links, and other components use an optimistic replication technique complemented by a background gossip protocol to ensure eventual consistency. In case of failure of one controller instance, the remaining ones can take mastership of the devices without loss of information. This approach, together with an advanced multi-thread software architecture, improves also the system scalability because the mastership of devices is typically balanced among the several instances [36]. Those reliability and scalability properties have been widely tested in emulated and demonstrative experimental scenarios [35, 36], and confirmed with the deployment of real networks such as a commercial deployment by China Unicom, and a worldwide deployment on a federation of research networks (e.g., Internet2, GÉANT, etc.) [37].

However, the multi-instance architecture of ONOS is designed in such a way the controller instances should be located within the same Local Area Networks, due to stringent bandwidth and latency requirements. For this reason, several projects and exemplar platforms using ONOS are considering the uti-

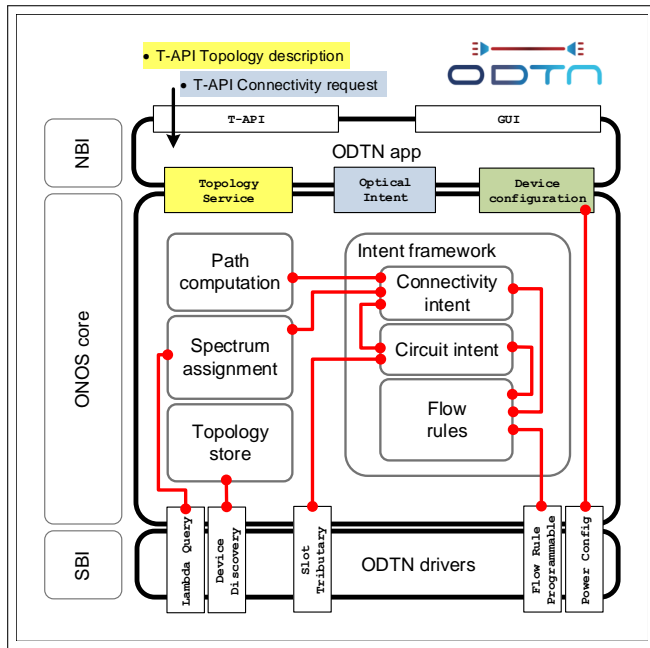


Fig. 2. ODTN reference architecture, most of the development work is concentrated on NBI and SBI.

lization of a hierarchy of (multi-instance) controllers where each controller is dedicated to a geographically distributed or different technological domains (e.g., electronic and optical). For instance, the E-CORD project (i.e., Enterprise CORD), which is in advanced phase of investigation by many worldwide operators including TIM [15], is using two different ONOS controllers at each Central Office (CO) (i.e., one for the access network and the other for CO fabric), and a distinct ONOS controller for the network interconnecting the COs, i.e., a multi-layer network including an optical layer. All those controllers are then coordinated by a parent ONOS interfaced with the network orchestrator (i.e., XOS in the CORD project). Another example is the control plane architecture designed within the METRO-HAUL project [3], where there is an ONOS controller at each metro node (i.e., devoted to control the local fabric **mainly composed of OpenFlow switches**), an ONOS controller for the optical metro/regional disaggregated optical network **connecting the nodes** and a parent controller **that keeps a summarized vision of the child networks and exposes** the overall network view to the orchestrator [38].

The intent-based organization of the ONOS controller is also a key feature for the deployment in optical networks. Essentially, the intent framework allows programmers of applications to specify high-level policies without worrying about low-level device details, moreover it ensures that the policy is met by transparently re-compiling the intents as a consequence of environment changes [39]. This feature, is used in case of network failures, after failure detection and localization the controller identifies the intents affected by the failure and, for each affected intent, triggers the operations required for the recovery **trying to minimize the re-configuration in the data plane**. Specifically, **within the current ONOS implementation, the recovery path is selected minimizing the number of devices to be re-configured in the data plane**.

4. ODTN WORKING GROUP

While most of the use cases and deployments mentioned in Sec. 3 are focused on the utilization of electronic devices and OpenFlow protocol, the utilization of ONOS as a controller of optical networks is still in the development phase with a number of experimental demonstrations that have been recently performed in collaboration between service providers, ONF and the research community [3].

The first attempt to enable ONOS to control an optical network has been made through the extended version of the OpenFlow protocol [7]. However, the utilization of the OpenFlow protocol has been demonstrated to be not **flexible enough** to encompass all the configurations required on the optical plane. In the meanwhile, several modeling initiatives started using YANG for the complete characterization of optical devices and configuration procedures [10, 11, 13]. With this kind of modeling language, the natural protocol to control and monitor the network became NETCONF [8, 40], that has also the strength of not requiring protocol extension to carry information on new parameters or features, only requiring an update of the model, and enables therefore effective experimentation of new features and technological evolution.

In this scenario, the Open and Disaggregated Transport Networks (ODTN) working group has been created at the ONF at the beginning of 2018 with the specific target to extend the ONOS functionality to configure and monitor disaggregated optical networks [41]. The target was also the utilization of standard interfaces and protocols on both sides of the controller, i.e., NorthBound (NBI) and the SouthBound (SBI). Specifically, the proposed approach is the utilization of T-API on the northbound interface and NETCONF/YANG on the southbound [9].

A. ODTN reference architecture

From the architectural point of view ODTN is mainly extending the ONOS controller on the NBI and SBI interfaces, as illustrated in Fig. 2. On the NBI, an ODTN app has been developed to expose a T-API interface, and most recently to implement a web-based graphical user interface (GUI) for direct interaction with the optical devices. Specifically, the NBI interface is already well-defined adopting T-API (version 2.0). The interface is implemented through RESTCONF protocol [42] and supports: the reception of connectivity service requests, and the upload/download of the network topology description into/from the distributed topology store. Within the ONOS core, the adoption of the intent framework facilitates the re-utilization of a set of already available ONOS features such as (elementary) RSA algorithms.

A connectivity service request can be issued between two client ports of a source and a destination transponder, or between two line ports. In the former case, the connectivity request is managed by the intent framework mapped into two separate intents, i.e., an *optical connectivity* intent and an *optical circuit* intent. In the latter case, only the optical connectivity intent is configured.

The optical connectivity intent encompasses the configuration of the optical layer (i.e., line side of the transponders and cross-connection of the traversed ROADMs or OLS). Thus, after the execution of path computation and spectrum assignment, the optical connectivity intent is typically translated into a number of *flow rules*: one rule for each traversed ROADM and one rule for the source and the destination transponders. If an OLS is present a single rule is generated and translated into a T-API con-

nectivity service to be forwarded to the OLS on the SBI. ROADM and OLS rules include input port, output port and the optical channel to be used. Transponder rules include the line port and the optical channel to be used.

The optical circuit intent encompasses the association of the transponders client port to the transponder line port utilized for the associated optical connectivity intent. Thus, the circuit intent is mapped into two flow rules, one for the source transponder and one for the destination transponder. The rules include the client port, the line port, and the mapping information. For instance, considering the OTN hierarchy, if the line port is *ODU2* (i.e., 10 Gbps) and the client ports are *ODU0* (i.e., 1.2 Gbps), the rule has to specify in which of the eight possible data units the client traffic has to be mapped into the line frame.

Once the intents have been compiled into a number of flow rules, those rules are forwarded to the device specific drivers on the SBI (i.e., to the *flow rule programmable* behaviour of the driver). A set of drivers have been developed and other are in phase of development *within* ODTN, supporting both proprietary and standard YANG models. Specifically, a different driver will be necessary for each type of devices (e.g., transponders, ROADMs, OLS) and for each considered YANG model (e.g., OpenConfig, OpenROADM, T-API). The driver essentially translates the flow rules into *NETCONF/OpenConfig-OpenROADM* or *RESTCONF/T-API* messages to be forwarded to the device for applying the required configuration.

Besides the aforementioned flow rule programmable behaviour, the drivers typically implement other behaviours, each one for employing a specific functionality. The *device description discovery* is called when the device is pushed in the topology to load the device details, such as the number and the types of ports. The *lambda query* behaviour is used during the spectrum assignment to retrieve the list of optical channels supported on each port. The *tributary slot* behaviour is used to perform effective multiplexing from transponder client ports into transponders line ports (i.e., mapping multiple circuit intents into a single connectivity intent). The *power config* behaviour is used by applications to retrieve information regarding target and current optical power information from each port of the devices.

Finally, besides the work focused on the NBI and SBI interfaces some aspects more inside the ONOS core have been extended for supporting specific requirements of optical devices. These extensions include: support of multi-instance ONOS deployment with NETCONF based devices; and support of uni-directional ports of optical devices, support of devices of type OLS.

B. ODTN phases

The ODTN work has been planned in three main phases, following the disaggregation levels as reported in Sec. 1. Phase 1.0 has been mainly dedicated to the development of the T-API based NBI while on the SBI, the work started considering the configuration of the transponder's line side. The phase 1.0 work has been demonstrated during 2018 in two independent environments: at NTT communication laboratories and at Telefonica laboratories. Considering both environments the tested devices include NEC, Infinera, Nokia and Cassini transponders [43].

In phase 1.5, an OLS has been included within transponder pairs so that the controller can start to be used in networks deployed on the field. A driver for configuration of the OLS has been developed and released that communicates with the OLS using T-API on the SBI. Phase 1.5 work has been experimentally demonstrated in [44].

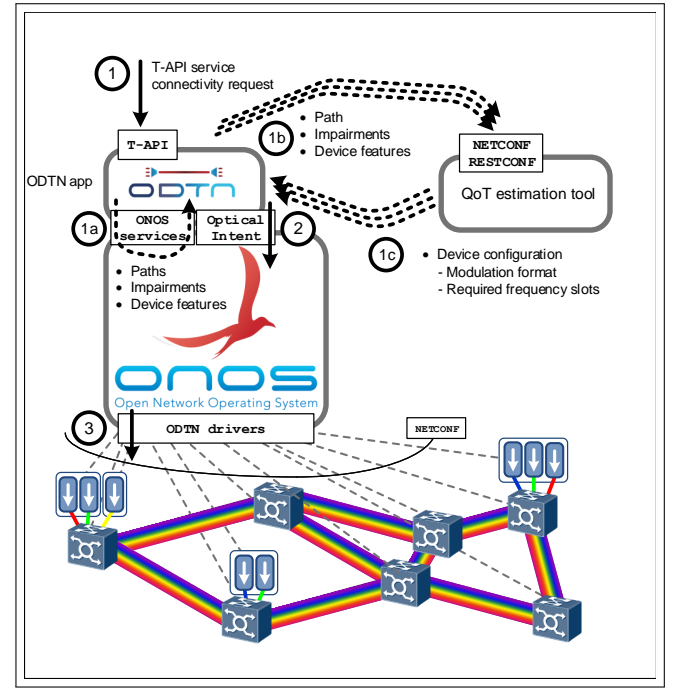


Fig. 3. SDN reference architecture and procedures to configure a lighthouse, current ODTN approach in solid lines, support of external planning and QoT estimation tools in dotted lines.

Phase 2.0 is currently on-going. In this phase the support of ROADMs will be introduced to enable the control of meshed networks, thus the development of drivers for the configuration of ROADMs using standard modeling is progressing. The configuration of Lumentum ROADMs and of emulated and experimental OpenROADM devices have been already demonstrated in [15, 31]. Also, in this phase the intent framework as described in Sec. 4A has to be accurately tested and extended to support more advanced configuration features, such as configuration of transmission power, forward error correction (FEC), modulation formats and support of flexible grid. For this purpose, the flow rule programmable and the power config behaviours will also require extensions, or new dedicated behaviours could be added to the drivers.

The plan for phase 3.0 is to go further disaggregating the ROADMs in more elementary devices such as, node degrees, filters, optical amplifiers, etc. However, since complete disaggregation could overload the controller without providing great benefit in terms of efficiency, it is still unclear up to which levels of disaggregation phase 3.0 will lead.

5. OPEN POINTS

Besides the on-going development of the aforementioned software components, the ONOS extension for enabling the effective control of optical disaggregated networks poses other open aspects to be considered, that are not yet explicitly included in the ODTN roadmap.

A. Integration with external tools

A first important requirement is to facilitate the integration of ONOS with external tools implementing advanced network planning, optical physical impairment modeling, data plane monitoring and telemetry. These features could be also imple-

mented inside the controller itself, but the current orientation of the community is to keep them external for two main reasons: do not overload the controller and foster a more modular architecture where this kind of tools can be independently developed and maintained by developers and researchers with specific expertise. Moreover, almost all network vendors and operators would continue to use their own tools to plan and provision the network (e.g., implementing advanced RSA algorithms). Finally, there are some well-established open-source initiatives in this space, where a lot of implementation work has been done that would be inefficient to replicate for porting those features inside the controller. Among these open-source initiatives is worth mentioning the Net2Plan [planning tool](#) [29] for which some integration with ONOS already started within the METRO-HAUL project; and the GNpy QoT estimation tool [26, 45] modeling non-linear optical impairments supported by the TIP consortium.

For the integration with external [QoT estimation tools](#) we propose the modular integration model depicted in Fig. 3, a similar model can be used for [external planning](#), monitoring and telemetry tools [24]. Fig. 3 reports in solid lines the current procedure implemented in ONOS: when a connectivity request is received from the T-API interface (step 1), the intent framework is utilized to map the request into a number of flow rules (step 2, see Sec. 4A for details), then the rules are translated into NETCONF configuration messages and sent to the data plane by using the specific drivers (step 3). The proposed work-flow to integrate an external QoT estimation tool (e.g., to evaluate the QoT on a set of candidate paths) is depicted in Fig. 3 with dashed lines. Specifically, the application on the NBI is extended to execute steps 1a, 1b, and 1c before invoking the intent framework. In step 1a, the application utilizes the needed ONOS services for computing the set of candidate paths to be evaluated, retrieving also the information for estimating the physical impairments (e.g., fiber types, fiber lengths) and information about the features supported by the devices (e.g., modulation formats, transmission power). Step 1b is executed for each computed path, specifically the paths are forwarded to the external tool using for instance RESTCONF or NETCONF protocols. For each path, the external tool provides a reply (step 1c) including the required details on the device configuration (e.g., modulation format to be used at the transponders, number of frequency slots to be reserved on the ROADMs). Then, based on the received information, the application selects the best path (e.g., the path where the highest spectral efficiency can be reached) and triggers the intent deployment. To this extent, the intent framework has to be extended to take into account the data provided by the external tools and consequently deploy the network connectivity.

For the integration of external planning and optimization tools [29] the T-API interface already provides the possibility to retrieve a detailed list of currently established intents (e.g., used path and spectrum assignment). But the same interface has to be extended to accept the results of the optimization (e.g., a new path and spectrum assignment for some of the established intents) to be submitted to the intent service to deploy the selected re-configurations.

B. Data plane configuration work-flow

A further requirement of optical networks, where devices are characterized by a significant configuration time, is the ability of the controller to know when a lightpath (i.e., an optical connectivity intent) has been completely configured on the data plane. At this regard, the utilization of NETCONF protocol is a

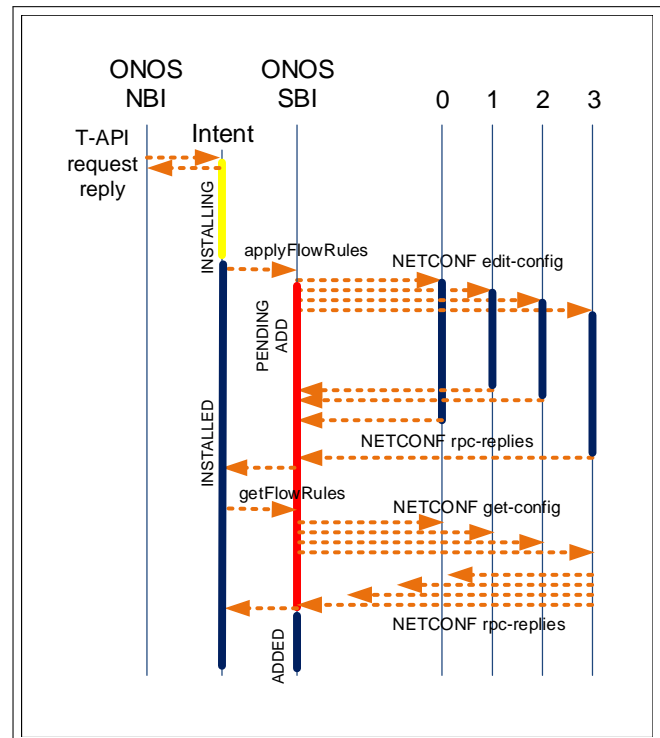


Fig. 4. Current protocol interactions between ONOS controller and the optical data plane. Controller immediately provides a reply to the T-API request without waiting for successful establishment; Intent is passed to Installed without waiting for successful establishment; Flow Rules are passed into the ADDED state only at next getFlowRules cycle.

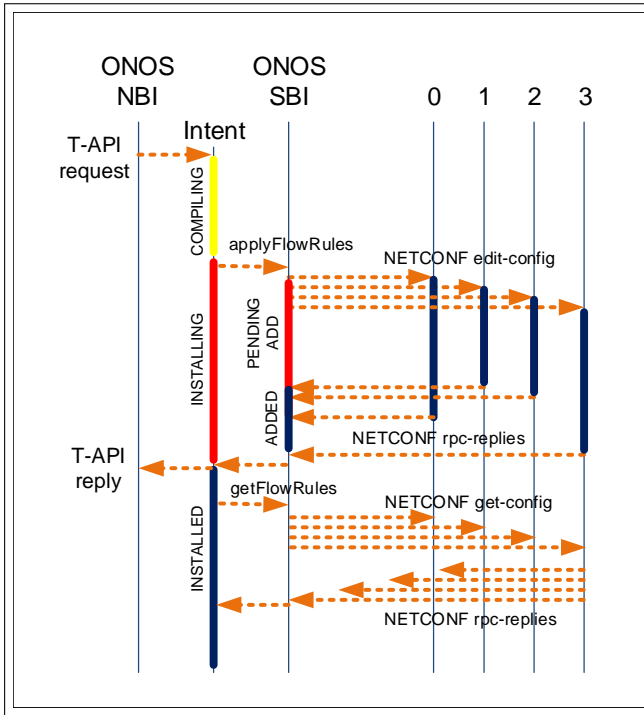


Fig. 5. Required protocol interactions between ONOS controller and the optical data plane. Controller provides a reply to the T-API request after successfully establishment; Intent is passed to Installed after successfully establishment; Flow Rules are passed into the ADDED state upon reception of NETCONF reply.

good starting point because it features a confirmation message for each configuration request. However, in the current ONOS implementation, see Fig. 4, the controller considers the intent as installed immediately after the execution of the RSA algorithm, without waiting that all flow rules have been successfully installed on the device (i.e., before confirmation the rules are kept in state PENDING ADD). Thus, the T-API reply is issued to the orchestrator when the lightpath is still not functionally established. Finally, the flow rules are turned in the ADDED state only after a whole monitoring cycle (i.e., typically every 20-30 seconds), thus ignoring the NETCONF reply arriving from the device after the application of the device configuration.

The desired behavior is instead depicted in Fig. 5: the controller replies to the T-API request only when data plane is totally configured, the intent is set as installed in the controller only when all the flow rules are in the ADDED state at the controller. Finally, each flow rule is set in the ADDED state immediately after the reception of the NETCONF confirmation message from the device.

6. EXPERIMENTAL DEMONSTRATION

The reported experiments evaluate the performance of the ONOS controller, including the main components of the ODTN SBI, and a real data plane deploying real optical and packet-based devices. The considered use-cases include: setup of end-to-end connectivity service transporting real traffic; traffic recovery using the features provided by the intent framework of ONOS considering a node failure on the optical layer.

The experimental testbed is described in Fig. 6. On the NBI

we have extended the ONOS REST interfaces to enable POST, GET and DELETE operations with optical intents. Moreover, the post operation has been extended so that a suggested path and optical channel can be included in the intent request, the suggested resources are then used by the intent only if currently available on the network. On the SBI the controller uses the OpenFlow drivers for HP commercial switches that we have developed and are now part of the master ONOS branch. Moreover, we are using the mentioned NETCONF drivers from ODTN (i.e., OpenConfig and OpenROADM drivers), and a set of drivers for commercial Lumentum ROADM devices that we have developed and contributed to ONOS.

On the data plane the testbed includes both real and emulated devices. On the Ethernet network layer, two commercial HP-3800 switches are included and configured by the controller using OpenFlow protocol. Emulated and real devices deployed on the optical layer include a device agent based on ConfD tool [14, 46] that has been implemented and deployed to sustain the NETCONF communication with the controller. Then, for real optical devices the NETCONF agent translates the NETCONF RPCs received from the controller in actual proprietary commands. Specifically, the testbed includes two transponders modeled following the OpenConfig YANG definition (i.e., Ericsson SPO with line cards at 10Gbps and 100 Gbps), and seven ROADMS (i.e., 4 emulated using OpenROADM model, 2 MEMS-based 1x4 switches, 1 Finisar WSS 1x4, and 1 Lumentum ROADM-20).

A. Experiment description

The experiments report on two use cases to evaluate the performance of the system in two different phases: *connectivity provisioning* and *connectivity restoration*. More specifically, the provisioning use case considers the setup of an end-to-end connectivity between two hosts involving the configuration of both packet-based and optical devices; the restoration use case considers the disruption of the previously established connectivity due to the failure of an optical device.

The targets of both experiments are: (i) register the sequence of events processed by the controller with the corresponding timing on both the packet and the optical layer; (ii) monitor the optical layer configuration; (iii) monitor the end-to-end connectivity configuration. The two performed experiments are described below.

1. *Connectivity provisioning*: Host-1 connected the SDN switch 1 starts sending ICMP echo requests (i.e., one ping requests every 20 ms). The SDN switch forwards the packets received on port 11 to the controller within a OpenFlow PackIn messages. Upon reception of the first PackIn message the controller triggers the configuration of both the optical data plane and the SDN switches. Specifically, an optical circuit intent is submitted from client port 2 of Transponder-1 to client port 1 of Transponder-2; and four OpenFlow rules are submitted to properly redirect traffic among SDN switch ports (e.g., in SDN switch 1 two rule are installed to redirect traffic from port 11 to port 9 and viceversa). When all configurations are correctly deployed on the network ICMP request/reply flow will regularly work between Host-1 and Host-2. The number of ICMP requests without reply is used as a measure of the end-to-end connectivity configuration time.
2. *Dynamic recovery with ROADM reconfiguration*: this use case starts with the end-to-end connectivity already established

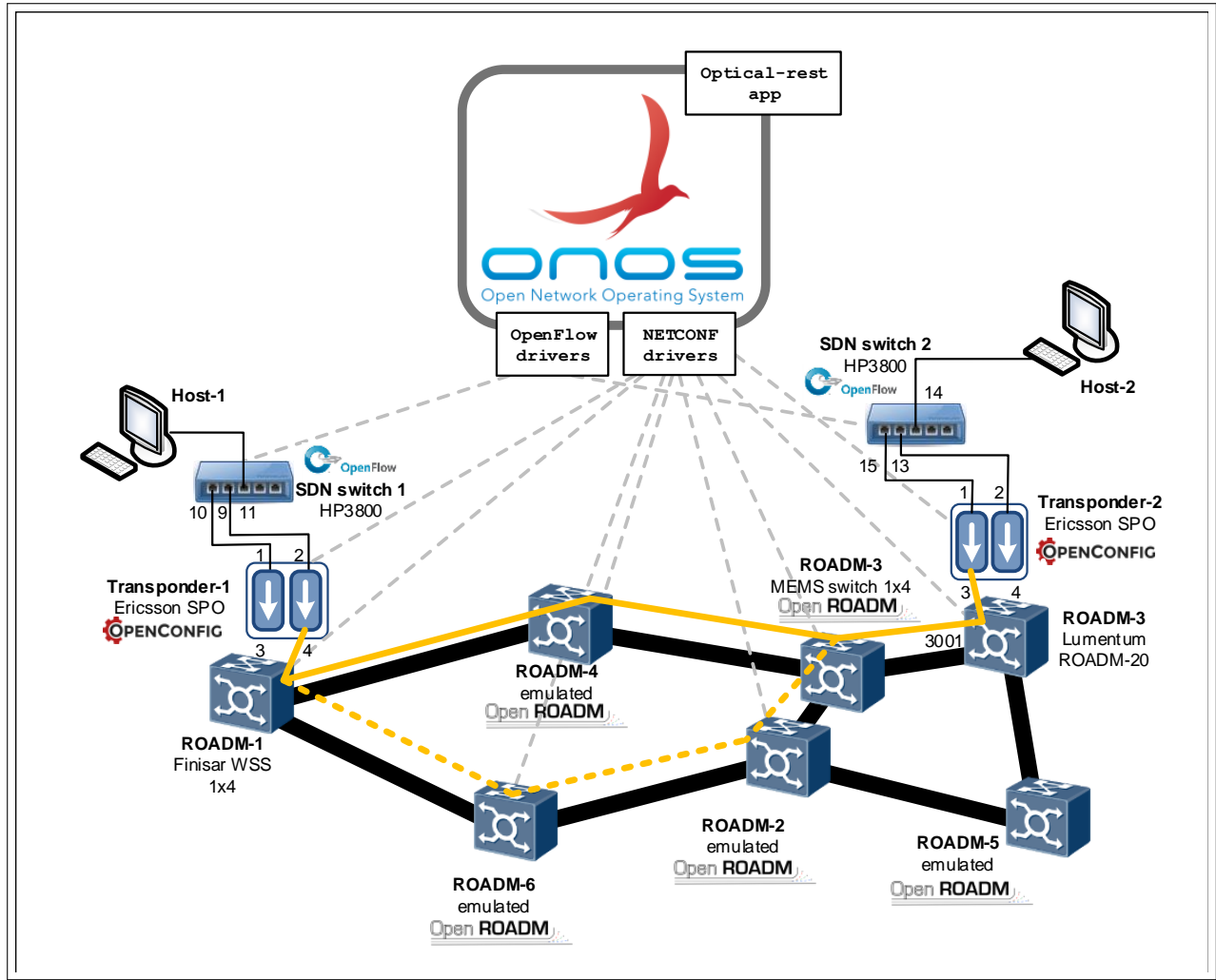


Fig. 6. Multi-layer experimental testbed.

between Host-1 and Host-2, along the solid yellow path illustrated in Fig. 6, and an active ICMP echo request/reply flow (i.e., one ping requests every 20 ms). A node failure is emulated by contextually turning off the NETCONF agent of ROADM-4 and disrupting the fiber between ROADM-1 and ROADM-4. The controller consequently re-configures the optical connectivity intent using the dashed yellow path. During the reconfiguration the ping flow is disrupted, the number of ICMP requests without reply is used as a measure of the end-to-end connectivity recovery time.

B. Experiment results

Fig. 7 and Fig. 9 report the events time-line of the two considered use-cases. The timing of the control plane events is provided by the ONOS logger, while the timing of data plane events has been measured using a differential approach starting from a common reference. An event is triggered on the data plane (e.g., ping is started from an host) that is detected on the control plane with low latency (e.g., corresponding PackIN is received at the controller).

1. *Connectivity provisioning*: Fig. 7(a) and (b) report the events time-line as logged by ONOS on the packet layer and on the optical layer, respectively. Fig. 7(c) reports the optical layer

configuration time showing the optical power received on port 3001 of the ROADM-3 (i.e., Lumentum ROADM-20), while Fig. 7(d) reports the packet layer configuration time showing the instant on which the first ping reply is received at Host-1. The sequence of events is in accordance with the work-flow reported in Fig. 4, where the intents are considered installed before a confirmation is received for the installation of the flow rules on the devices. The experiment has been repeated 2120 times, all events showed some time variance where rules requests have been always generated within 100 ms (50 ms in Fig. 7), intents have been always installed within 2 seconds (1 s in Fig. 7), OpenFlow rules have been always added within 5 seconds (2 s in Fig. 7), the new link has been always detected within 20 seconds (11 s in Fig. 7), and NETCONF rules have been always added within 30 seconds (9 s in Fig. 7). Finally, the time distribution for optical layer and packet layer configuration time obtained over the 2120 experiments is reported in Fig. 8 where: the average time to complete the configuration of the optical layer is 1358 milliseconds, i.e., 1 s in Fig. 7(c); the average time to complete the configuration of the packet layer is 5033 milliseconds, i.e., 6 s in Fig. 7(d). Thus, we can assert that the physical configuration of optical layer takes about 26.9% of the total time needed to have an operational

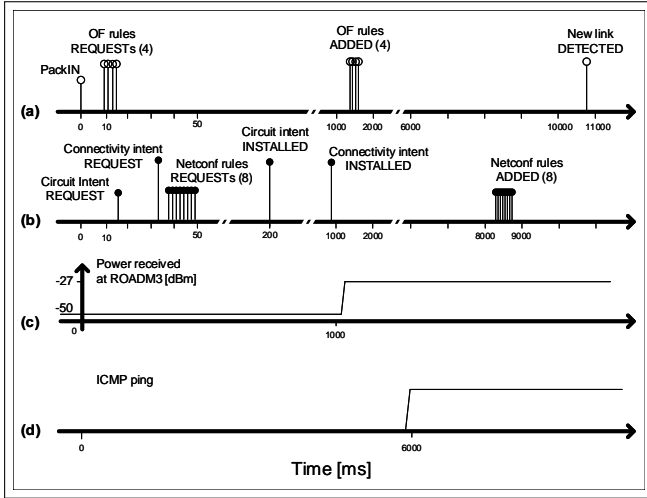


Fig. 7. Use case 1 time-line of events: control plane events on packet layer (a); control plane events on optical layer (b); data plane events on optical layer (c); data plane events on packet layer (d).

packet layer connectivity.

2. *Dynamic recovery with ROADM reconfiguration:* Fig. 9 reports the events time-line as logged by ONOS and measured on the data plane. Contextually with the emulated failure, Host-1 sends an ICMP request that is forwarded to the controller within an OpenFlow PackIN message, whose reception is assumed as reference of the failure time. The experiment has been repeated 10 times. The node failure is detected at the controller through timeout of the NETCONF session in about 1.5 – 2.5 seconds; the node failure detection causes the change of the optical connectivity intent state to CORRUPT, REQUEST and finally INSTALLED with contextual generation of modified NETCONF rules to be sent to involved devices. Specifically, in the considered case modified rules are sent to ROADM-1, ROADM-6, ROADM-2 and ROADM-3 to activate the dashed yellow path in Fig. 6. This phase is typically completed within 1 seconds after failure detection. Then the NETCONF rules are turned in the ADDED state within 30 seconds. **The average time to complete the configuration of the optical layer is 2793 milliseconds, and the average time to complete the configuration of the packet layer is 6568 milliseconds.** The link down/up events on the packet layer registered around time 8-10 seconds in the experiment reported in Fig. 9 are related to the values of the timers used for generating LLDP packets, they have been always detected between 8 and 20 seconds after the failure, in some experiments the link down has been never detected. **The time distribution for optical layer and packet layer re-configuration time obtained over the 10 experiments is reported in Fig. 10 where: the average time to complete the re-configuration of the optical layer is 2793 milliseconds, i.e., about 3 seconds in Fig. 9(c); the average time to complete the re-configuration of the packet layer is 6568 milliseconds, i.e., about 7.5 seconds in Fig. 9(d).** Thus, we can assert that the physical configuration of optical layer takes about 42.5% of the total time needed to have an operational packet layer connectivity.

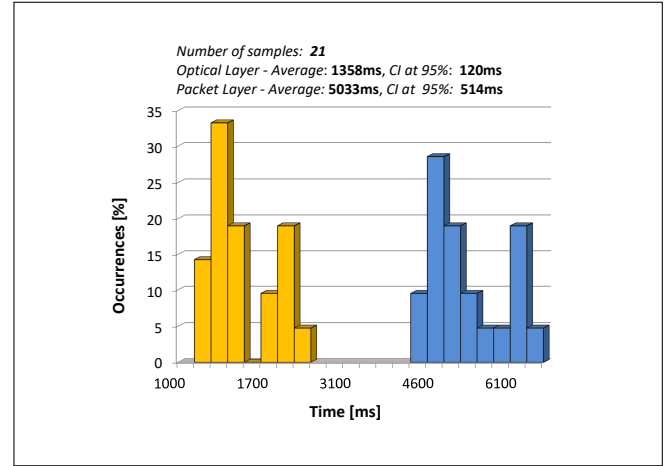


Fig. 8. Use case 1 time distribution of data plane events. Yellow bars refers to optical layer setup; cyan bars refers to packet layer setup. **For both measures the achieved confidence interval, at 95% of confidence level, is around 10% of the average value.**

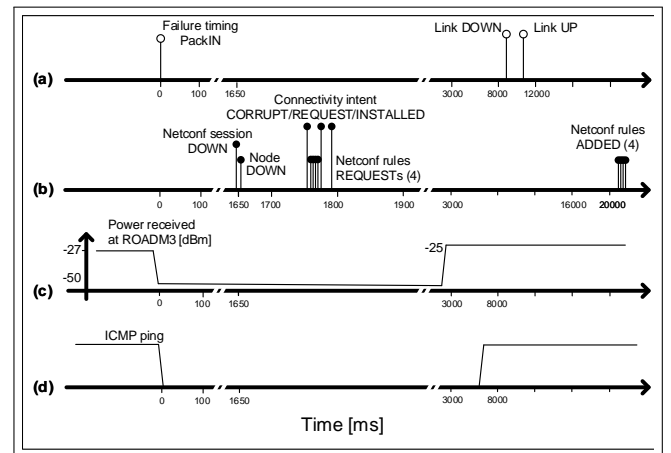


Fig. 9. Use case 2 time-line of events control plane events on packet layer (a); control plane events on optical layer (b); data plane events on optical layer (c); data plane events on packet layer (d).

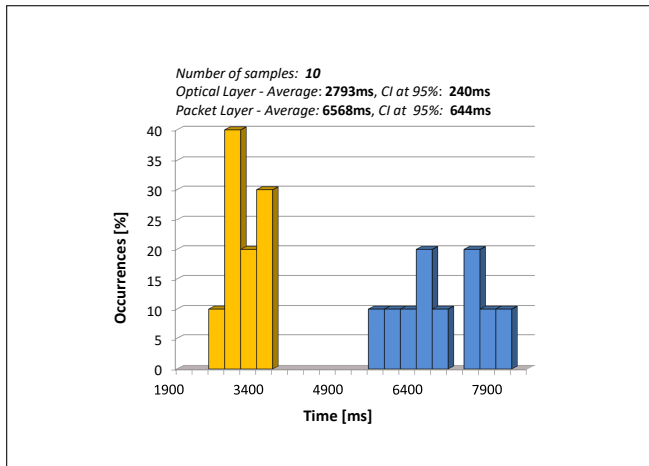


Fig. 10. Use case 2 time distribution of data plane events. Yellow bars refers to optical layer recovery; cyan bars refers to packet layer recovery. For both measures the achieved confidence interval, at 95% of confidence level, is below 10% of the average value.

C. Experiment results discussion and future work

The time for the connectivity provisioning registered in the experiments (use case 1) can be considered a good achievement because it is not a critical requirement and because it has to be compared against the time needed to setup a new channel in traditional transport networks managed by proprietary systems, that is typically measured in hours or days.

Conversely, the time registered for traffic recovery (use case 2) is not satisfactory, and mechanisms to significantly reduce the recovery time are needed. As a first step, a fast procedure to detect the failure on the data plane is required at the controller, i.e., waiting for the expiration of the NETCONF connection timeout is not an option. On this direction, one possibility is to extend the controller to react to NETCONF notification that can be sent by the transponder agents when a loss-of-light is detected on a line port (similar mechanisms are supported by ONOS for packet switching device using OpenFlow protocol). However, reducing the detection time is not enough, because the modification of the lightpath route (implying reconfiguration of traversed ROADMs) disrupts the point-to-point connectivity for several seconds, see time interval between node failure detection in Fig. 9(b) and first ping reply received after failure in in Fig. 9(d). Thus, the utilization of dedicated protection mechanisms, e.g., replicating the optical connectivity of two disjoint paths, is the most viable solution to provide fast traffic recovery in case of failures on the optical layer. To this extent, as future work, the ONOS controller should be extended with a new type of intent (i.e., protected connectivity intent) that, in conjunction with a fast failure detection mechanism on the optical layer, would be able to provide sub-second traffic recovery.

Another important aspect to be accurately assessed before practical utilization of ONOS in real optical network deployments is the scalability of the configuration and re-configuration time in case of realistic number of established lightpaths. Again, this is especially critical for the recovery use case, where besides the risk of controller overload the possible contentions for the re-configurations of the data plane devices should be taken into account. From this point of view, the utilization of a centralized controller can be also an opportunity to be exploited because, as

reported in [34], it introduces the possibility to coordinate the recovery process and aggregating the required re-configuration messages.

7. CONCLUSION

After an overview of previous work on SDN-controlled optical networks, this paper reported on ONOS potentials, performance and limitations for enabling control and monitoring of disaggregated optical networks, with specific focus on the on-going work within the ODTN working group at ONF.

Detailed experimental results are reported considering two relevant use-cases: the connectivity setup and the recovery of the connectivity in case of failure events on the optical layer. The utilized ONOS controller includes the main components (e.g. OpenConfig and OpenROADM drivers) that are in phase of development within ODTN. On the data plane, besides emulated optical devices, the utilized experimental testbed includes also real optical and packet-switched devices.

The obtained experimental results demonstrated that the connectivity setup is performed in a satisfactory time interval (i.e., few seconds). Conversely, the recovery time, that has much more strict requirements, needs to be significantly improved before deployment on the field.

FUNDING

Work funded by the EC H2020 project METRO-HAUL (761727).

ACKNOWLEDGMENT

We thank the members of the ONF ODTN and METRO-HAUL projects for their insightful comments and suggestions concerning the OpenConfig ONOS driver implementation.

REFERENCES

1. S. Gringeri, N. Bitar, and T. J. Xia, "Extending software defined network principles to include optical transport," *IEEE Commun. Mag.* **51**, 32–40 (2013).
2. J. Santos, N. Costa, and J. Pedro, "On the impact of deploying optical transport networks using disaggregated line systems," *J. Opt. Commun. Netw.* **10**, A60–A68 (2018).
3. R. Casellas, R. Martínez, R. Vilalta, and R. Muñoz, "Metro-haul: Sdn control and orchestration of disaggregated optical networks with model-driven development," in *2018 20th International Conference on Transparent Optical Networks (ICTON)*, (IEEE, 2018), pp. 1–4.
4. E. Riccardi, P. Gunning, Ó. G. de Dios, M. Quagliotti, V. López, and A. Lord, "An operator view on the introduction of white boxes into optical networks," *J. Light. Technol.* **36**, 3062–3072 (2018).
5. Open Networking Foundation, "OpenFlow Switch Specification," <https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf>.
6. Open Networking Foundation, "Optical Transport Protocol Extensions," https://www.opennetworking.org/wp-content/uploads/2014/10/Optical_Transport_Protocol_Extensions_V1.0.pdf.
7. S. Yamashita, A. Yamada, K. Nakatsugawa, T. Soumiya, M. Miyabe, and T. Katagiri, "Extension of openflow protocol to support optical transport network, and its implementation," in *2015 IEEE Conference on Standards for Communications and Networking (CSCN)*, (IEEE, 2015), pp. 263–268.
8. T. Kunz and K. Muthukumar, "Comparing openflow and netconf when interconnecting data centers," in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, (IEEE, 2017), pp. 1–6.
9. M. De Leenheer, Y. Higuchi, and G. Parulkar, "An open controller for the disaggregated optical network," in *2018 International Conference*

- on *Optical Network Design and Modeling (ONDM)*, (IEEE, 2018), pp. 230–233.
10. OpenROADM MSA, “OpenROADM MSA Device White Paper v2.2,” <http://www.openroadm.org>.
 11. Open Config, “Open Config data models,” <http://openconfig.net>.
 12. M. Filer, M. Cantono, A. Ferrari, G. Grammel, G. Galimberti, and V. Curri, “Multi-vendor experimental validation of an open source qot estimator for optical networks,” *J. Light. Technol.* **36**, 3073–3082 (2018).
 13. Open Device, “Infinera Open Device models,” <https://github.com/infinera/OpenDevice>.
 14. A. Sgambelluri, A. Giorgetti, F. Paolucci, P. Castoldi, and F. Cugini, “Open source implementation of openconfig telemetry-enabled netconf agent,” in *ICTON*, (2019).
 15. R. Morro, F. Lucrezia, P. Gomes, R. Casellas, A. Giorgetti, L. Velasco, E. Riccardi, A. C. Piat, A. Percelsi, J. Pedro *et al.*, “Automated end to end carrier ethernet provisioning over a disaggregated wdm metro network with a hierarchical sdn control and monitoring platform,” in *2018 European Conference on Optical Communication (ECOC)*, (IEEE, 2018).
 16. “Opendaylight,” <https://www.opendaylight.org/>.
 17. “Open network operating system (onos),” <https://onosproject.org/>.
 18. Open Networking Foundation, “ONF Transport API,” <https://www.opennetworking.org/wp-content/uploads/2017/08/TAPI-2.0-Updates-Overview.pdf>.
 19. Y.-D. Lin, C.-C. Wang, C.-Y. Huang, and Y.-C. Lai, “Hierarchical cord for nfv datacenters: resource allocation with cost-latency tradeoff,” *IEEE Netw.* pp. 1–7 (2018).
 20. A. Giorgetti, R. Casellas, R. Morro, A. Campanella, and P. Castoldi, “Onos-controlled disaggregated optical networks,” in *Tech. Dig. OFC*, (IEEE/OSA, 2019).
 21. R. Alvizu, G. Maier, N. Kukreja, A. Pattavina, R. Morro, A. Capello, and C. Cavazzoni, “Comprehensive survey on t-sdn: Software-defined networking for transport networks,” *IEEE Commun. Surv. & Tutorials* **19**, 2232–2283 (2017).
 22. L. Gifre, J.-L. Izquierdo-Zaragoza, M. Ruiz, and L. Velasco, “Autonomic disaggregated multilayer networking,” *J. Opt. Commun. Netw.* **10**, 482–492 (2018).
 23. L. Velasco, A. Sgambelluri, R. Casellas, L. Gifre, J.-L. Izquierdo-Zaragoza, F. Fresi, F. Paolucci, R. Martínez, and E. Riccardi, “Building autonomic optical whitebox-based networks,” *J. Light. Technol.* **36**, 3097–3104 (2018).
 24. F. Paolucci, A. Sgambelluri, F. Cugini, and P. Castoldi, “Network telemetry streaming services in sdn-based disaggregated optical networks,” *J. Light. Technol.* **36**, 3142–3149 (2018).
 25. M. Shiraiwa, N. Yoshikane, S. Xu, T. Tsuritani, N. Miyata, T. Mori, M. Miyabe, T. Katagiri, S. Yoshida, M. Tanaka *et al.*, “Experimental demonstration of disaggregated emergency optical system for quick disaster recovery,” *J. Light. Technol.* **36**, 3083–3096 (2018).
 26. P. Poggiolini, G. Bosco, A. Carena, V. Curri, Y. Jiang, and F. Forghieri, “The gn-model of fiber non-linear propagation and its applications,” *J. lightwave technology* **32**, 694–721 (2013).
 27. V. Lopez, O. G. de Dios, and J. P. Fernandez-Palacios, “Whitebox flavors in carrier networks,” in *Optical Fiber Communication Conference*, (2019).
 28. F. Pederzoli, M. Chamania, M. Santuari, T. Szyrkowicz, C. Matrakidis, C. Rozic, D. Klonidis, V. Lopez, and D. Siracusa, “Disaggregating optical nodes in a multi-layer sdn orchestrator for the integration of an in-operation planning tool,” in *Optical Fiber Communication Conference*, (2018).
 29. M. Garrich, F.-J. Moreno-Muro, M.-V. B. Delgado, and P. P. Mariño, “Open-source network optimization software in the open sdn/nfv transport ecosystem,” *J. Light. Technol.* **37**, 75–88 (2019).
 30. J. Kundrat, O. Havlis, J. Jedlinsky, and J. Vojtech, “Opening up roadms: Let’s build a disaggregated open optical line system,” *J. Light. Technol.* (2019).
 31. A. Sgambelluri, J.-L. Izquierdo-Zaragoza, A. Giorgetti, L. Gifre, L. Velasco, F. Paolucci, N. Sambo, F. Fresi, P. Castoldi, A. C. Piat *et al.*, “Fully disaggregated roadm white box with netconf/yang control, telemetry, and machine learning-based monitoring,” in *Optical Fiber Communication Conference*, (Optical Society of America, 2018), pp. Tu3D–12.
 32. A. Triki, C. Betoule, G. Thouenon, O. Henais, N. Pelloquin, G. Lambert, M. Sylla, S. Vachhani, D. Bhardwaj, M. Dorfman *et al.*, “Openroadm compliant sdn controller for a full interoperability of the optical transport network,” in *2018 European Conference on Optical Communication (ECOC)*, (IEEE, 2018), pp. 1–3.
 33. F. Pederzoli, M. Gerola, A. Zanardi, X. Forns, J. F. Ferran, and D. Siracusa, “Yamato: the first sdn control plane for independent, joint, and fractional-joint switched sdm optical networks,” *J. Light. Technol.* **35**, 1335–1341 (2017).
 34. A. Giorgetti, F. Paolucci, F. Cugini, and P. Castoldi, “Dynamic restoration with gmpls and sdn control plane in elastic optical networks,” *J. Opt. Commun. Netw.* **7**, A174–A182 (2015).
 35. A. S. Muqaddas, P. Giaccone, A. Bianco, and G. Maier, “Inter-controller traffic to support consistency in onos clusters,” *IEEE Transactions on Netw. Serv. Manag.* **14**, 1018–1031 (2017).
 36. A. Bianco, P. Giaccone, R. Mashayekhi, M. Ullio, and V. Vercellone, “Scalability of onos reactive forwarding applications in isp networks,” *Comput. Commun.* **102**, 130–138 (2017).
 37. ONF, “Onos project wikispaces, global sdn deployments powered by onos,” <https://wiki.onosproject.org/display/ONOS/Deployments>.
 38. R. Casellas, R. Martínez, R. Vilalta, and R. Muñoz, “Control, management, and orchestration of optical networks: Evolution, trends, and challenges,” *J. Light. Technol.* **36**, 1390–1402 (2018).
 39. D. Sanvito, D. Moro, M. Gulli, I. Filippini, A. Capone, and A. Campanella, “Onos intent monitor and reroute service: enabling plug&play routing logic,” in *2018 4th IEEE Conference on Network Softwareization and Workshops (NetSoft)*, (IEEE, 2018), pp. 272–276.
 40. R. Enns, “Network configuration protocol netconf,” *IETF RFC 6241* (2011).
 41. “Open disaggregated transport network,” <https://onosproject.org/odtn>.
 42. A. Bierman, “Restconf protocol,” *IETF RFC 8040* (2017).
 43. “Open disaggregated transport network,” <https://wiki.onosproject.org/display/ODTN/Demos>.
 44. A. Campanella, H. Okui, A. Mayoral, D. Kashiwa, O. G. de Dios, D. Verchere, Q. P. Van, A. Giorgetti, R. Casellas, R. Morro *et al.*, “Odn: Open disaggregated transport network. discovery and control of a disaggregated optical network through open source software and open apis,” in *Optical Fiber Communication Conference*, (Optical Society of America, 2019), pp. M3Z–4.
 45. “Gnpy,” <https://github.com/Telecominfraproject/oopt-gnpy>.
 46. “ConfD, management agent software framework for network elements,” <https://www.tail-f.com/confd-basic/>.

AUTHOR BIOGRAPHIES

Alessio Giorgetti received the Ph.D. degree from Scuola Superiore Sant’Anna (SSSA), Pisa, Italy, in 2006. In 2007, he has been visiting scholar at Centre for Advanced Photonics and Electronics, University of Cambridge, UK. Currently, he is an Assistant Professor at SSSA. His research interests include: optical network architecture and control plane, industrial network design, software defined networking. He is author of more than 100 publications including international journals, conference proceedings, and patents.

Andrea Sgambelluri received the Ph.D. from Scuola Superiore Sant’Anna, Pisa, in 2015. In March 2015 he won the grand prize at 2015 OFC Corning Outstanding Student Paper Competition with the paper: “First Demonstration of SDN-based Segment Routing in Multi-layer Networks”. In 2016 he was postdoc researcher KTH Royal Institute of Technology (Optical Networks Laboratory (ONLab)). Currently, he is postdoc researcher at the TeCIP

Institute of Scuola Superiore Sant'Anna, Pisa, Italy. His main research interests are in the field of control plane for packet and optical networks, including SDN/NFV, segment routing and YANG/NETCONF.

Ramon Casellas (SM'12)

graduated from UPC, Barcelona and from the ENST, Paris, in 1999, with a Erasmus student exchange program grant. He completed a PhD degree in 2002 and worked at the ENST as an Associate Professor. In March 2006, he joined the CTTC Optical Networking Dept., where he is currently holding a Senior Researcher position. His research interest areas include Traffic Engineering and Control and Management of Transport Networks, including the GMPLS/PCE architecture, Software Defined Networking (SDN) and Network Function Virtualization (NFV). He has been involved in several international, national and industrial research projects and has co-authored 5 book chapters, over 200 journal and conference papers and 5 IETF RFCs.

Roberto Morro received a Dr.

Ing. degree in electronic engineering from University of Genoa (Italy) in 1988. After a six year experience with Marconi as a testing engineer, he joined TIM (at that time CSELT) in 1995 where he is currently working in the technology evolution and innovation unit. He has been actively working in several European funded projects and in the related technology transfer inside the TIM group. His research interests include network control, SDN, NFV, focusing especially on multi-layer (IP over optics), multi-domain and traffic engineering aspects.

Andrea Campanella is

employed as a Member of Technical staff (MTS) at Open Networking Foundation (ONF). He started working with ONF through an internship in October 2015. Andrea is currently the leader of the Optical Disaggregated Transport Network (ODTN) Reference Design and Exemplar Platform. Andrea participated in OFC 18 with 2 invited talks on optical disaggregation and Intent Based Networking, bringing also a demo about end to end optical orchestration through ODTN. Andrea also is the bridge between the ONF ODTN community and TIP's OOPT group. Together the two teams already demonstrate white-box, open source controlled optical solutions at different conferences, ONF connect, OFC'19 and OCP'19. Andrea also sits on the ONOS Technical Steering Team.