JOURNAL OF
Optical Communications
and Networking

Enabling hierarchical control of coherent pluggable transceivers in SONiC packet–optical nodes

Alessio Giorgetti,[1,*] Davide Scano,[2] Andrea Sgambelluri,[2] Francesco Paolucci,[3] Emilio Riccardi,[4] Roberto Morro,[4] Piero Castoldi,[2] and Filippo Cugini[3]

[1]CNR-IEIIT, Pisa, Italy
[2]Scuola Superore Sant'Anna, Pisa, Italy
[3]CNIT, Pisa, Italy
[4]TIM, Turin, Italy
*alessio.giorgetti@cnr.it

Received 10 October 2022; revised 30 January 2023; accepted 1 February 2023; published 23 February 2023

Traditional metro networks are composed of packet switching nodes (i.e., routers) interconnected by optical transport links. In this scenario, the packet and optical domains are clearly separated, using dedicated controllers. Standalone muxponders/transponders will be replaced in optical metro and transport networks by the utilization of hybrid packet–optical nodes equipped with coherent pluggable transceivers. Thus, the traditional packet control plane is unable to manage and fully support the specific optical parameters required to configure such pluggable modules. Moreover, the coordination between the optical and packet layers within this hybrid node has not been standardized yet and requires careful design to enable effective management of connectivity services. This paper proposes two software-defined networking (SDN)-based hierarchical solutions to coordinate and control coherent pluggable transceivers in a multi-layer network exploiting hybrid packet–optical nodes. The two solutions have been designed and validated focusing on the pluggable module configuration and the communication within the SDN hierarchical architecture. An experimental testbed including two packet–optical nodes, running an extended version of the open-source Software for Open Networking in the Cloud (SONiC) operating system, is deployed to show the effectiveness of the two solutions, with a deep analysis of the time required to set up end-to-end connections spanning the packet and optical domains. © 2023 Optica Publishing Group

https://doi.org/10.1364/JOCN.477732

1. INTRODUCTION

The recent evolution of transmission technology has driven the introduction of pluggable transceivers provided with a coherent detection strategy [1–4]. For example, digital coherent optics (DCO) transceivers are commercially available at rates of 100, 200, and 400 Gb/s in different form factors, such as C form factor pluggable type 2 (CFP2) and quad small form factor pluggable double density (QSFP-DD). The CFP2 transceiver (55 cm^3, 41.5 mm width, 107.5 mm length, 12.4 mm thickness) is suitable for metro and long-haul interconnections; relying on 7 nm technology, it provides a signal launch power of 0 dBm and is able to traverse multiple reconfigurable optical add–drop multiplexers (ROADMs). Experiments have shown the capability to cover up to 1500 km at 400 Gb/s in a 75 GHz spaced dense wavelength-division multiplexing system, using 16 quadrature amplitude modulation at 69 Gbaud, probabilistic constellation shaping, and soft-decision forward error correction [5]. In addition, excellent interoperability performance has already been achieved by DCO-CFP2 based on an OpenROADM multi-source agreement [6,7]. The 400ZR coherent pluggable module based on the smaller QSFP-DD form factor (14 cm^3, 18.35 mm width, 89.4 mm length, 8.5 mm thickness) currently provides a signal launch power of up to −10 dBm, and it is mainly designed for data center interconnections, covering single span distances of up to 120 km. Better performance in terms of transmission power and optical reach can nowadays be obtained using 400ZR+ proprietary versions. For example, new 400ZR+ QSPF-DD products have been successfully validated over 1000 km relying on an output power of around 0 dBm [8]. The new coherent modules will represent an extremely attractive solution to be equipped directly within packet switching devices (e.g., those designed for the wider data center market). This would drive the reduction/removal of transponders and muxponders as standalone network elements particularly in metro networks, leading to a number of remarkable benefits: (i) reduced capital expenditure; (ii) reduced footprint; (iii) reduced latency, avoiding passing through an intermediate element; (iv) reduced

1943-0620/23/030163-11 Journal © 2023 Optica Publishing Group
Authorized licensed use limited to: Scuola Superio Sant Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

power consumption by avoiding the optical–electronical–optical (OEO) interconnection between gray router interfaces and colored transponder line interfaces (overall savings in the range between 50 and 150 W for every 100 Gb/s of nominal traffic, depending on the line rate).

Furthermore, coherent pluggables enable tight integration between packet and optical networks, which is of special interest as transport is dominated by Ethernet and IP traffic. For example, a single packet switch can serve as the spine in a leaf and spine architecture for intra-data center traffic aggregation, and, due to coherent pluggables, it can also provide effective interconnection of data centers. Despite the fact that not all transmission scenarios are suitable for pluggable modules (e.g., ultra long-haul), the interest around hybrid packet–optical nodes encompassing coherent pluggables is growing remarkably.

In the meantime, the Optical Internetworking Forum (OIF) is successfully leading the definition of the Common Management Interface Specification (CMIS) with related extensions for coherent modules, Coherent CMIS (C-CMIS) [9]. CMIS is becoming the management interface of choice for pluggable modules. It provides a well-defined mechanism to initialize and manage optical (and copper) modules in a standard way, while still guaranteeing the capability to support custom functionalities. Work is ongoing to support CMIS and C-CMIS within the operating systems of packet–optical boxes. Software for Open Networking in the Cloud (SONiC) is an open-source network operating system (NOS) already widely deployed in production intra-data center networks, and it is also considered a strong candidate to control packet–optical nodes. Indeed, SONiC provides an abstract interface called a switch abstraction interface (SAI) that guarantees a vendor-independent way to control the packet-switching layer of network devices. Since SONiC also supports CMIS and C-CMIS for the configuration of optical modules, it currently provides the perfect environment for network operators to develop hardware-agnostic networking applications involving both the packet and optical layers. SONiC provides a set of default applications; however, extensions are needed to fulfill operator requirements. For example, SONiC does not natively support NETCONF, and it does not fully encompass all the needed software components to operate on coherent pluggable transceivers.

In the context of ongoing modeling and standardization activities, OpenConfig and OpenROADM represent the most relevant initiatives for disaggregated architectures [10]. Both initiatives are addressing the support of hybrid packet–optical nodes by defining specific YANG models for NETCONF-based agent implementations [11]. Considerable work on disaggregation is also ongoing in the Telecom Infra Project (TIP) [12] and in the Open Disaggregated Transport Network (ODTN) initiatives [10]. For example, TIP drove the design on the first generation of open packet–optical nodes (named Cassini), providing a flexible mix of 100 GbE packet switching ports and 100/200 Gb/s coherent pluggable interfaces, and is currently involved in developing its SONiC-based NOS (i.e., Goldstone) for hybrid packet–optical nodes.

In the context of the scientific literature, so far most of the work on partial and full disaggregation has focused on transmission modules as standalone network elements, such as transponders and muxponders [13–17]. Besides the aforementioned technological limitations related to transmission and power consumption as well as the full support in NOSs, there is another challenge to taking full advantage of coherent pluggables: the coordinated configuration of packet and optical parameters within the same node. Indeed, those configurations are typically provided by two different software-defined networking (SDN) controllers, one in charge of packet resources and the other in charge of optical transport. So far, this aspect has been discussed only in our previous works [18,19] and in [20].

This paper expands upon the aforementioned works in three directions: (i) reporting on the reference scenario and related works; (ii) providing a detailed description of the applications developed on top of the SDN controller; (iii) deploying two solutions to enable coordinated SDN control of packet–optical nodes equipped with coherent pluggables on a common testbed, thus enabling a detailed experimental comparison.

2. REFERENCE SCENARIO AND PROBLEM DESCRIPTION

Figure 1 shows a traditional metro network using packet nodes (i.e., routers) attached to standalone transponders and interconnected through optical line systems (OLSs), typically composed of a number of ROADMs and optical amplified fiber links. The transponders provide OEO conversion between the client signal (short reach) originated by the routers and the line signal (long reach) crossing the OLS with adequate transmission performance. In this scenario, the SDN architecture is implemented with a clear separation of domains. Three controllers are typically considered: a hierarchical SDN controller (HrC) coordinating the end-to-end connectivity, an optical SDN controller (OptC) in charge of the transponders and OLS (e.g., adopting OpenROADM and OpenConfig models [11]), and a packet SDN controller (PckC) in charge of the packet switching devices. Traditionally, each controller has full and exclusive visibility on all the components and software modules of the underlying network elements. For example, OptC is the unique entity accessing the transponders, while PckC is the unique entity configuring the packet nodes. In addition, to not overload OptC, the idea of optical network digital twins is emerging in the research community, consisting of a dedicated control plane element that is aware of all optical physical impairments and interacts with OptC to provide feedback regarding the optical parameters to be used [e.g., transmit (TX) power, modulation format, etc.] [21].

The introduction of packet–optical nodes imposes the redesign of the overall SDN control architecture. Indeed, transponders are replaced by packet–optical nodes equipped with pluggable modules, and the traditional control mechanisms provided by PckC are not sufficient to configure the optical parameters. Since, in large metro networks, a single controller with visibility of both layers is not feasible due to scalability issues, proper coordination among involved controllers needs to be defined to enable successful establishment of end-to-end connectivity services. For example, without proper information exchange among controllers, PckC cannot know which

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

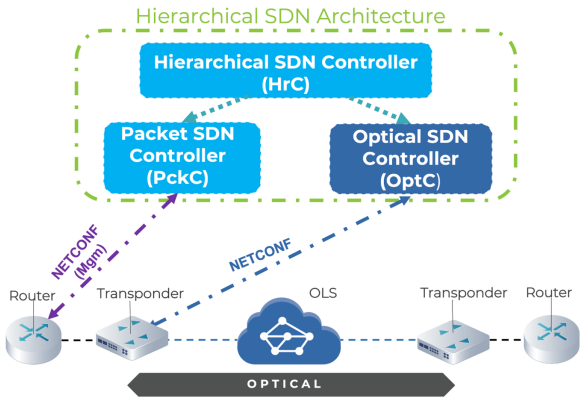

Fig. 1. Traditional SDN architecture for transponder-based optical networks.

wavelengths are available in the optical domain, and therefore it is not able to properly tune pluggable modules. Conversely, OptC is not aware of the wavelengths supported by the pluggables (i.e., tunability range), and therefore it is not able to choose the wavelength to reserve on the optical domain. The same applies to other optical parameters, such as optical power levels and operational modes.

3. PROPOSED SOLUTIONS FOR COORDINATED CONTROL OF PLUGGABLE MODULES

This section details the two considered solutions (see Fig. 2) to provide coordinated control of pluggable modules equipped within packet–optical nodes. Since network operators typically use NETCONF for controlling optical devices, this protocol is proposed also for managing SONiC-based devices, not requiring operators to fully rethink their control plane architecture.

The first approach, here named Exclusive (Excl), is shown in Fig. 2(a). The Excl approach provides access to packet–optical nodes from PckC only. That is, configurations related to both packet forwarding and optical pluggables are enforced by PckC. In this case, the optical parameters (e.g., central frequency, TX power, operational mode) have to be initialized at OptC, e.g., imported from PckC through HrC. The NETCONF protocol is considered for the pluggables configuration, while NETCONF or programming protocol-independent packet processor (P4) protocols [22,23] can be considered for the packet layer configuration.

The second approach, here named Shared (Shar), is shown in Fig. 2(b). The Shar approach relies on the joint control of packet–optical nodes from both PckC and OptC. Configurations related to packet forwarding are provided by PckC, while those related to optical pluggable modules are enforced directly by OptC. In this case, proper solutions are needed to guarantee coordinated access as well as the proper control segregation for avoiding possible conflicts. As in the previous approach, the NETCONF protocol has been considered for the pluggable modules configuration, while packet configuration can be performed via NETCONF or P4 protocols.

The two solutions are detailed in the following by illustrating the workflow adopted for establishing an end-to-end connectivity service (i.e., intent) traversing both the packet and optical domains. The concept of intent is typical of the SDN environment, where the central controller can enforce high-level policies without worrying about low-level device details. Moreover, intents generalize the concept of connectivity because they ensure that the target policy is met by allowing dynamic reconfiguration as a consequence of network changes (e.g., in the case of link failure) [10,24].

A. Configuration Workflow Using the Exclusive Approach

The workflow for the establishment of an end-to-end intent using the Excl approach is depicted in Fig. 3. At step 1, when HrC receives a connectivity request, it computes the sequence of domains and the edge nodes of each traversed domain. If the activation of a new lightpath is required, HrC sends an optical intent request to OptC (step 2). At step 3, OptC performs

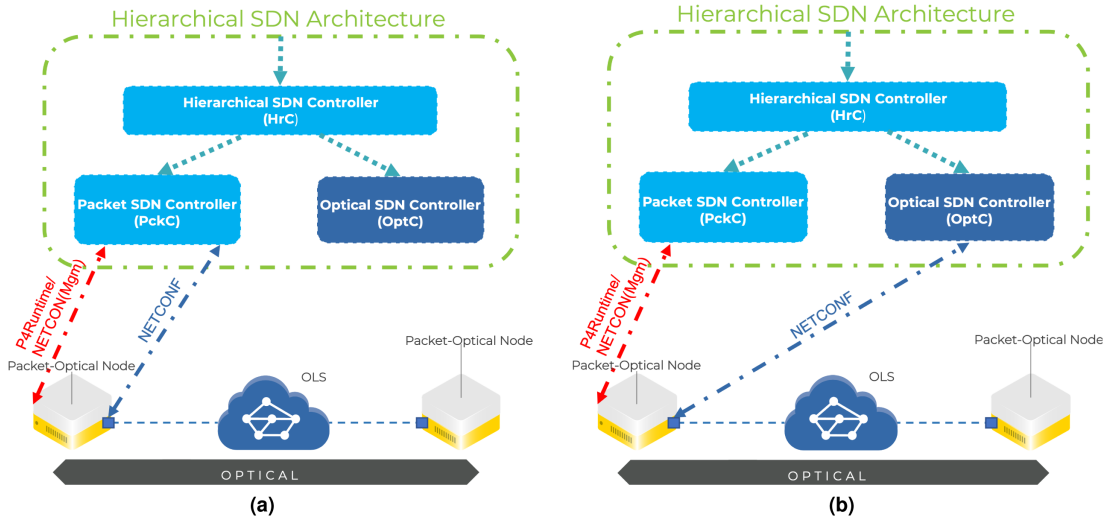

Fig. 2. Considered control and management scenarios. (a) Exclusive hierarchical control plane solution and (b) shared hierarchical control plane solution.

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

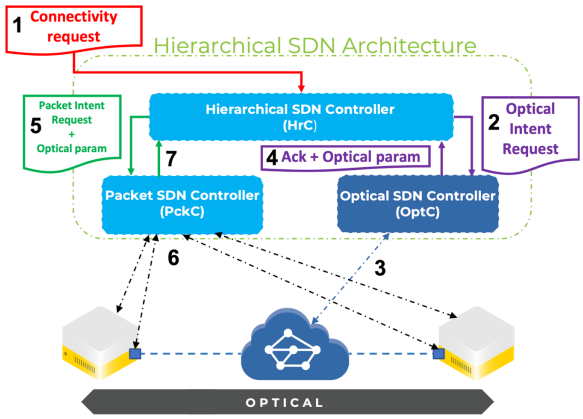

Fig. 3. Exclusive end-to-end intent setup workflow.

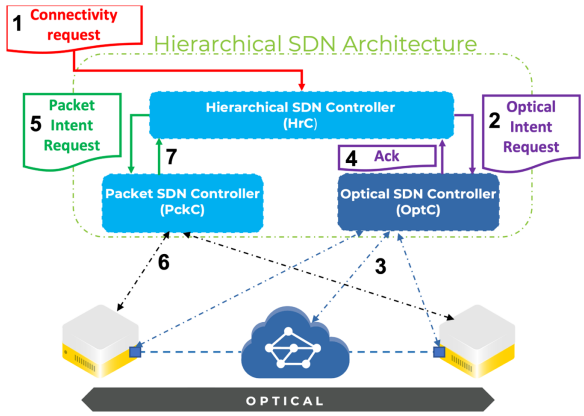

Fig. 4. Shared end-to-end intent setup workflow.

path computation and spectrum assignment, configures the OLS devices traversed by the lightpath, and, once the optical intent is installed, sends a reply to HrC, including the utilized optical parameters (step 4). At step 5, HrC shares with PckC the values notified by OptC (i.e., frequency, TX power, and operational mode) and requests the setup of a packet intent. PckC configures the pluggable modules, and, once the link becomes active, it installs the packet intent (step 6). Finally, at step 7, PckC informs HrC that the packet connection was successfully configured. If a path with enough bandwidth already exists in the optical layer, steps 2, 3, and 4 are skipped by HrC, directly moving to step 5 for the installation of the packet intent without requiring the activation of additional optical pluggables.

Using the Excl approach, a network initialization phase is required during which OptC becomes aware of the physical parameters (e.g., tunability range) supported by the pluggable modules and of the association between each pluggable module and the connected OLS port. Since this information is quasi-static, it can be initialized through a specific configuration, e.g., using representational state transfer application programming interfaces (REST APIs) of the controllers.

B. Configuration Workflow Using the Shared Approach

The workflow for the establishment of an end-to-end intent using the Shar approach is depicted in Fig. 4. At step 1, HrC receives a connectivity request, and computes the sequence of domains and the edge nodes of each traversed domain. If the activation of a new lightpath is required, HrC sends an optical intent request to OptC (step 2). At step 3, OptC performs path computation and spectrum assignment, configures the OLS devices and the pluggables involved in the lightpath, and, once the optical intent is installed, it sends back a notification to HrC (step 4). In this case, the configured optical parameters are not notified to the HrC, because the configuration of the optical domain is fully managed by OptC. At step 5, HrC requests PckC to configure a new packet intent. PckC installs the packet intent (step 6) and informs HrC that the packet connection was successfully configured (step 7). If a path with enough bandwidth already exists in the optical layer, steps 2, 3, and 4 are skipped by HrC, directly moving to step 5 for the installation of the packet intent.

4. ARCHITECTURE IMPLEMENTATION

Implementation of the two proposed solutions is based on four main contributions: (i) enhancement of the packet–optical node structure with the NETCONF agent and P4 dataplane, (ii) design and implementation of HrC, (iii) design and implementation of PckC, and (iv) design and implementation of OptC. The four contributions are described in the following sections.

A. Packet–Optical Node

The structure of the considered packet–optical node is shown in Fig. 5. An Ethernet switch runs the SONiC [25] operating system in the Open Network Install Environment (ONIE) [26]. In addition to the default SONiC applications (i.e., sonic-cfggen, syncd, swss, pmon, and the redis database), it may include containerized functionalities, i.e., the NETCONF agent container and the P4/P4Runtime container, depending on scenario requirements. The NETCONF agent container is used to configure and monitor optical pluggables. The agent uses the OpenConfig model for hardware representation, including ports and pluggables. To avoid misconfiguration issues when multiple controllers access the node, ownership segregation has been implemented using the Network Configuration Access Control Model (NCACM) solution, as detailed in RFC 8341. NCACM has been conceived to restrict NETCONF access to specific operations and contents in a user-based fashion. More specifically, for each configured user (i.e., PckC and OptC), a set of rules is configured in the NETCONF agent, permitting or denying operations (e.g., write, read-only) over specific prefix-based configuration parts.

In particular, considering the Shar approach in Fig. 2, OptC is provided with writing rights on optical parameters and read-only rights (including enabling notifications upon subscriptions) on packet parameters. Similarly, PckC is provided with writing rights on packet parameters and read-only rights

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

on optical parameters. In the Excl approach, access segregation is not required since only PckC accesses the node for configuration, with read and write permission.

The NETCONF container is able to directly access the C-CMIS driver of the physically connected pluggable modules, managing the optical parameters. In more detail, C-CMIS is formed by four software layers: (i) the physical interconnection layer manages the lower-level protocols needed to carry elementary signals from/to the module, (ii) the data transfer layer defines the data transmission protocols used on the physical interconnection layer, (iii) the register access layer provides the primitive operations to read and write bytes from/to a 256-byte addressable memory window, and (iv) the management application translates higher-level functions to the lower layers and is the interface utilized by the NOS applications toward the pluggable.

As we do not have coherent pluggable modules (e.g., ZR/ZR+ modules) in our laboratories, an external coherent transceiver (e.g., an Ericsson SPO xPonder with coherent 100G line ports) acts as a pluggable. That is, its configuration is not provided by the SDN controller directly; instead, it is provided by the packet–optical node as for locally equipped transceivers. Specifically, the muxponder is equipped with a custom REST server to configure and read the optical parameters, as shown in Fig. 5. In more detail, when the controller interacts via NETCONF with the packet–optical node, the agent leverages the REST connection to configure/read the optical parameters to/from the xPonder, as if it were a pluggable module attached to the box. The NETCONF agent container deployed on SONiC is able to access the C-CMIS driver of a pluggable card, if present. This way, the proposed internal design of the packet–optical node is ready to fully support optical pluggables.

The P4/P4Runtime is the interface leveraged by PckC to configure the packet layer; the P4 program configures the application-specific integrated circuit (ASIC), leveraging the lower-level APIs provided by the NOS, which are the software development kit and the SAI. P4/P4Runtime can be deployed on packet–optical nodes in different ways, i.e., using a P4 integrated network stack (PINS), Stratum, or proprietary solutions provided by switch vendors. PINS [27] is an application developed to bring the programmability of SDN into the SONiC environment, while Stratum [28] can be deployed on Open Network Linux [29] and Ubuntu, providing P4Runtime and

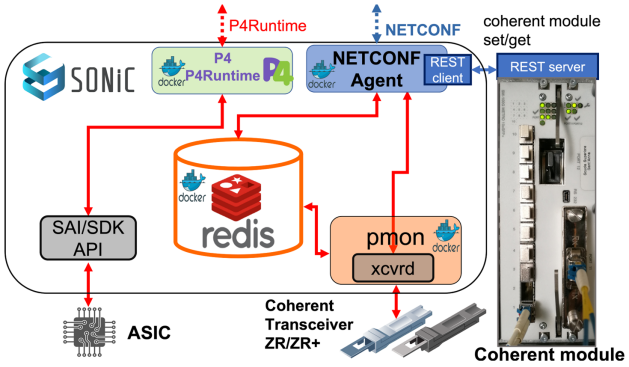

Fig. 5. Packet–optical node architecture: the packet–optical node exposes P4 runtime and NETCONF interfaces toward the control plane.

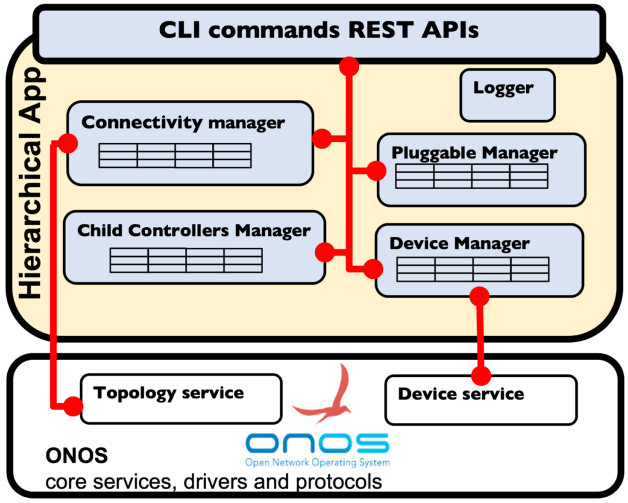

Fig. 6. Internal architecture of the Hierarchical App and its relations with ONOS core services.

OpenConfig interfaces. Both PINS and Stratum are available only on switches based on Intel or Broadcom ASICs. In our implementation, the P4/P4Runtime has been implemented using a containerized BMv2 switch [30] implementation. This container-based solution exploits all P4 capabilities (i.e., header manipulation, table customization, in-band telemetry, flow-rule programmability), but processing all traffic at software level.

B. Hierarchical SDN Controller

The implementation of HrC is based on the Open Network Operating System (ONOS) SDN controller [31] and required the development of a dedicated ONOS module, i.e., the Hierarchical App. Specifically, the architecture of the Hierarchical App is illustrated in Fig. 6. This application retrieves and maintains the information on the status of the entire network, communicating with child SDN controllers through REST APIs. The app encompasses four main functional blocks. The Child Controllers Manager module is in charge of discovering child controllers and locally storing their information (e.g., IP address). The Device Manager module stores links and devices of the entire network, knowing, for example, the child SDN controller associated with each device. The Pluggable Manager module stores the pluggables information, keeping track of the pair (device, port) where each pluggable is attached. The Connectivity Manager module handles connectivity requests. For each connection request, it computes the end-to-end path and splits the computed path between child controllers, relying on both the packet and optical domains; i.e., where needed, a new lightpath is installed in the optical domain. This module is also responsible for recording all connections installed in the entire network.

The Hierarchical App provides both REST APIs and a set of command line interface (CLI) commands. The REST interface exposes a set of custom calls needed for exchanging information with the child SDN controllers, e.g., link updates and optical parameters, as explained in Sections. 3.A and 3.B. The CLI is used to display the information of each device, link,

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

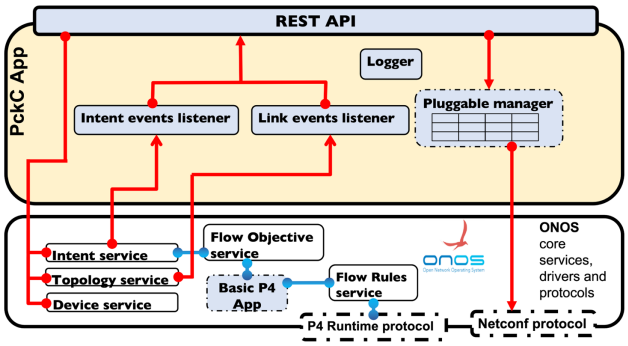

Fig. 7. Internal architecture of the PckC App and its relations with ONOS core services.

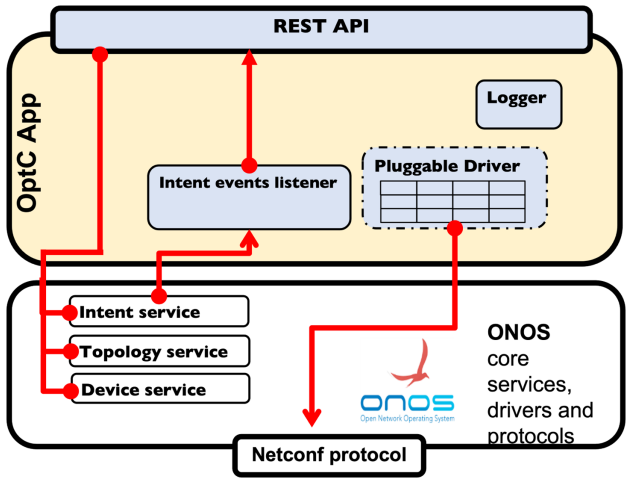

Fig. 8. Internal architecture of the OptC App and its relations with ONOS core services.

connection, and pluggable in the network. New connection requests can be requested to the Hierarchical App using either the REST or CLI interface.

C. Packet SDN Controller

Implementation of PckC is based on ONOS relying on the specifically designed PckC App, whose internal architecture is represented in Fig. 7. This application allows communication between PckC and HrC. Two versions of the application have been developed, considering the Excl and Shar approaches. In more detail, the PckC App may be deployed with (or without) the Pluggable Manager module and, alongside it, the Basic P4 App may be (or may not be) activated.

The Pluggable Manager module implements a NETCONF client that is used to retrieve and configure the pluggable's optical parameters. In addition to that, the module stores the interfaces on which the pluggables are placed. Intent Events Listener and Link Events Listener are modules in charge of notifying HrC in the case of intent installation or link changes (i.e., new link discovered or link failure).

The basic P4 App is an application available in the ONOS master repository, providing a P4 pipeline implementing basic packet forwarding. The pipeline is loaded and managed in the packet–optical node via P4Runtime protocol. All packet connection requests pass through the Basic P4 App module that converts each request into messages compliant with the pipeline.

HrC uses the REST API provided by ONOS to retrieve the network topology managed by PckC. In addition, the PckC App exposes a new REST API to set up an intent and/or configure/retrieve the pluggables parameters.

D. Optical SDN Controller

Implementation of the OptC is based on ONOS relying on the specifically designed OptC App, whose internal architecture is represented in Fig. 8. The OptC App enables communication between OptC and HrC, managing and retrieving the pluggables optical parameters. The OptC App is developed with or without the Pluggable Driver (dashed module in Fig. 8), according to the two considered scenarios, i.e., Excl does not use the Pluggable Driver.

The Pluggable Driver module implements a NETCONF driver to manage and control the pluggables in the packet–optical nodes, e.g., it converts the abstract configuration parameters computed within ONOS in NETCONF `edit-config` messages compliant with the OpenConfig model used to represent the pluggable modules. The Intent Event Listener module is used to notify HrC when the optical intent is successfully installed. The REST API provides a set of calls to request an optical connection, to notify the status of the lightpaths, and to get the optical parameters configured.

In addition, for supporting the Excl approach where OptC configures only ROADMs (while the configuration of the pluggables is performed by PckC), the ONOS intent service has been extended to support intent requests whose end-points are ROADM interfaces.

5. EXPERIMENTAL VALIDATION AND RESULTS

This section describes the testbed setup used for the experimental validation of the proposed solutions and the related results, collected considering the two workflows, i.e., Excl and Shar.

A. Testbed Description

The packet–optical testbed topology is illustrated in Fig. 9 and includes two packet–optical nodes equipped with pluggable transceivers, three emulated ROADMs (e.g., OpenROADM NETCONF agents running in dedicated docker containers), and four P4-based emulated switches (e.g., running a BMv2 software switch). Traffic is generated via Spirent N4U, placed at the edge of a packet network, as shown in Fig. 9. The P4 switches are implemented inside bare metal servers (Intel Xeon E5-2643 v3 6-core 3.40 GHz clock, 32 GB RAM) and connected with dedicated Ethernet interfaces using Mellanox ConnectX3 Network Interface Cards. Each packet–optical node consists of a Mellanox/NVIDIA SN2010 Ethernet switch, which runs the SONiC operating system over ONIE. In addition to the basic SONiC components, the P4/P4Runtime and NETCONF docker containers have

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

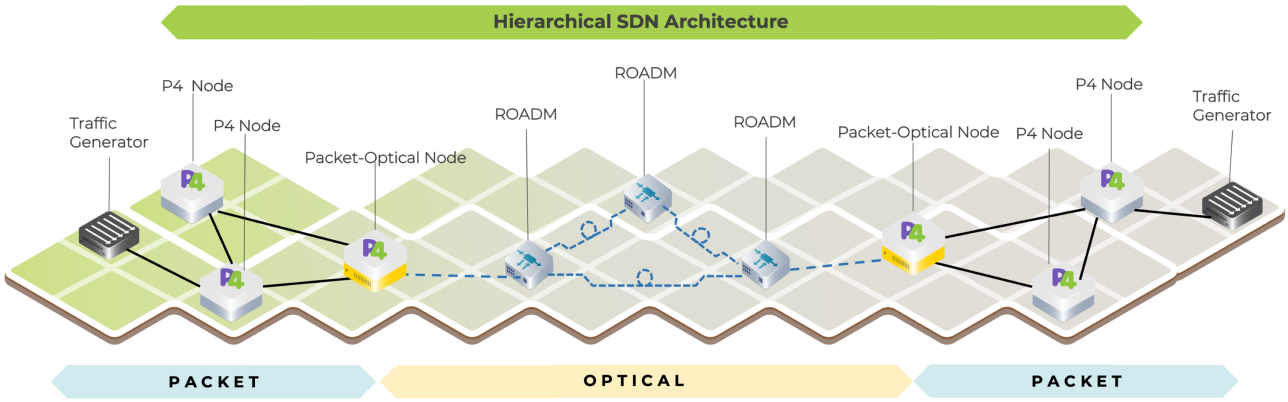

Fig. 9. Testbed topology.

been added, as shown in Fig. 5. As presented before, the xPonder Ericsson SPO with coherent line ports at 100G acts as a pluggable coherent module in packet–optical nodes.

The control plane is realized by three ONOS SDN controllers: PckC, OptC, and HrC. The two child controllers run on two workstations equipped with an Intel i7-8700 12-core 3.2 GHz clock, 32 GB RAM, while HrC runs on a dedicated workstation with an Intel Xeon W-2223 8-core 3.60 GHz clock, 16 GB RAM.

Considering the Excl approach, Fig. 2(a), PckC is in charge of configuring the packet domain via P4Runtime, and optical pluggables via NETCONF. OptC controls the ROADMs via NETCONF. Conversely, using the Shar approach, Fig. 2(b), PckC controls only the packet domain while the configuration of ROADMs and optical pluggables is left to OptC via NETCONF.

B. Results

The scenarios presented in the previous sections have been implemented and validated in the testbed illustrated in Fig. 9 starting from an empty network where no lightpaths are configured in the optical network; thus, the packet domain is composed of two islands. The first goal of the proposed experimental test is to validate the procedure to establish an end-to-end intent (i.e., spanning across the packet and optical domains). Other than the functional validation, the end-to-end connection setup time (i.e., T_{e2e}) has been measured and analyzed collecting several time contributions by logging the events registered at HrC. The main time contributions building up the end-to-end connection setup time are (i) T_{opt}, the optical intent setup time; T_{link}, the discovery time of the new link in the packet domain [this step is automatically achieved by PckC by means of the Link Layer Discovery Protocol (LLDP)]; and T_{pck}, the packet intent setup time.

1. Exclusive Workflow Experiments

In the experiments related to the Excl workflow, PckC is responsible only for the configuration of the packet–optical nodes, using NETCONF for the pluggables configuration and P4 for the packet forwarding setup.

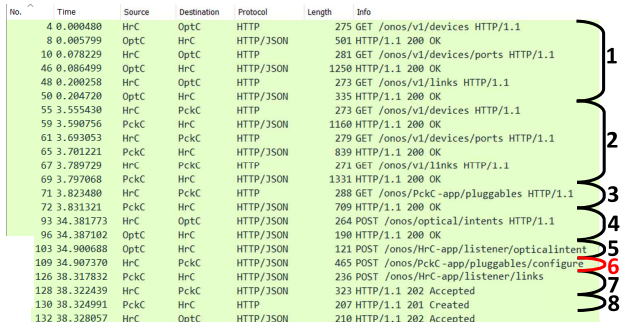

Fig. 10. Packets capture at HrC during Exclusive workflow execution.

The proposed solution has been functionally validated, analyzing the interactions among the three SDN controllers. The REST messages exchanged by HrC and the two SDN child controllers are captured in Fig. 10. All messages from 4 to 72 are exchanged during a preliminary phase during which HrC retrieves topological information from the child controllers by invoking three different REST GET methods. Specifically, the 6 messages from 4 to 50 (labeled 1 in Fig. 10) are used to discover the optical network topology; then, the packet network topology is retrieved with messages from 55 to 69 (label 2); finally, messages 71 and 72 are related to the pluggable details discovery (label 3). The setup of the optical intent is started with a REST POST message toward OptC (label 4). After path computation and resource assignment, OptC returns with message 103 including the optical parameters that will be needed to configure the optical pluggables (label 5). Thus, the time elapsed between message 71 and message 103 is equivalent to T_{opt}. Then, HrC uses a POST message toward PckC to configure the pluggables in the packet–optical nodes and to set up the packet intent (label 6). PckC replies with two events: when the link is discovered (label 7) and when the packet intent is finally installed (label 8). Thus, the time elapsed between message 109 and message 126 is equivalent to T_{link}, while the time elapsed between message 126 and message 130 is equivalent to T_{packet}.

Looking at the capture time stamps, the procedure in this case takes around 4 s to perform the end-to-end intent configuration spanning across the packet and optical domains. In

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

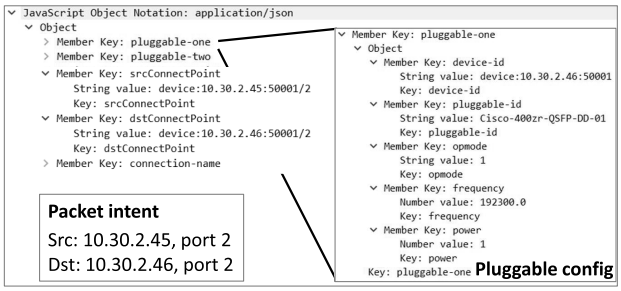

Fig. 11. Pluggables configuration message used in the Exclusive approach.

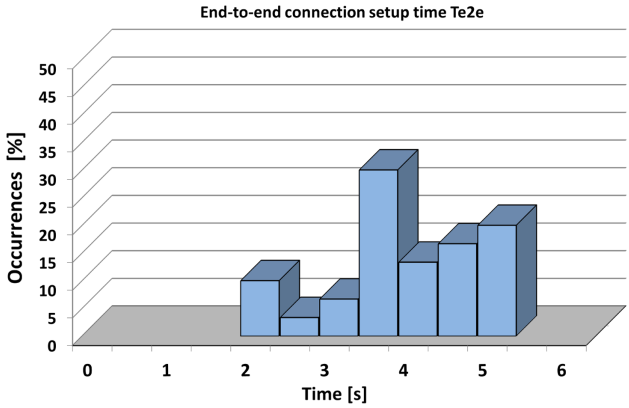

Fig. 12. Distribution of T_{e2e} over 30 experiments using the Exclusive workflow, average value $\bar{T}_{e2e} = 3.59$ s.

more detail, $T_{e2e} = 3.94$ s, $T_{\text{opt}} = 0.52$ s, $T_{\text{link}} = 3.41$ s, and $T_{\text{pck}} = 0.01$ s.

Figure 11 details the POST message sent by HrC to PckC (message 109 in the capture) to ask for the pluggables configuration and the activation of the packet intent. Such a message contains the following data: the device-id necessary for PckC to locate the pluggable to be configured and the pluggable-id and related optical parameters to be configured (i.e., operational mode, frequency, launch power). In addition, the message includes src-ConnectPoint and dst-ConnectPoint, identifying the packet intent end-points that are used to set up the packet intent, as soon as the two pluggables are configured and the activated link is detected between the two packet network islands.

Figure 12 presents the time distribution needed to complete the overall workflow (i.e., T_{e2e}). Specifically, the experimental configuration of an end-to-end intent has been repeated 30 times, where each experiment encompasses steps 4, 5, 6, 7, and 8 as reported in Fig. 10. The average value registered for the end-to-end setup time is 3.59 s (i.e., $\bar{T}_{e2e} = 3.59$ s). In addition, the histogram shows that more than 80% of the samples are distributed in the range from 3.5 and 5.5 s, without samples above 5.5 s.

Among the contributions listed above, the experimental data revealed that both the optical intent setup time and the packet intent setup time are negligible. Indeed, the former is distributed in the range from 0.52 to 0.61 s (accounting for the time needed for the requested intent to arrive in the state INSTALLED), while the latter ranges from 0.01 to 0.02 s.

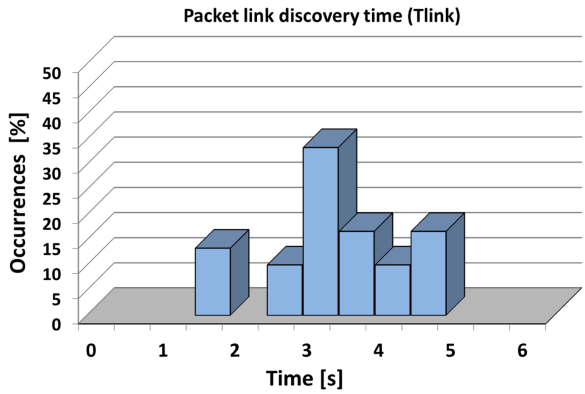

Fig. 13. Distribution of T_{link} over 30 experiments using the Exclusive workflow, average value $\bar{T}_{\text{link}} = 2.93$ s.

Figure 13 illustrates the distribution of the link discovery time T_{link} that has been demonstrated to be the most relevant contribution to T_{e2e}, i.e., on average, T_{link} counts for more than 80% of the overall T_{e2e}. In this case, the measured time includes the communication time between HrC and PckC, the time needed by PckC to apply the required configuration to the pair of optical pluggables in the two packet–optical nodes, the time for physical activation of the pluggables, and the time needed for the link discovery implemented by PckC by periodically sending LLDP packets. In this case, the average value registered for the end-to-end setup time is 2.93 s (i.e., $\bar{T}_{\text{link}} = 2.93$ s). As shown in the histogram, more than 80% of the samples are distributed in the range from 3 to 5 s and there is not any sample above 5 s. This result is in accordance with the rate of generation of LLDP packets applied by the ONOS controller that by default is 3 s.

2. Shared Workflow Experiments

In the experiments related to the Shar workflow, the packet and optical SDN controllers are both involved in the configuration of packet–optical nodes: OptC uses NETCONF to configure the ROADMs and pluggables, while PckC is responsible for the packet forwarding configuration via P4.

The proposed solution has been functionally validated, analyzing the interaction among the three SDN controllers and the REST messages exchanged by HrC and the two SDN child controllers as captured in Fig. 14. Messages from 111 to 439 are used to discover the optical network topology (label 1) from OptC, while messages from 581 to 815 are used to retrieve the packet network topology (label 2) from PckC. In this case, pluggables are discovered together with the other optical device details. Then, HrC requests OptC to set up an optical intent using the REST POST message (i.e., messages 938 and 977, label 3). In this case, OptC performs the complete optical configuration, including ROADMs and pluggable modules located in packet–optical nodes. Thus, the time elapsed between message 938 and message 977 is equivalent to T_{opt}. Once the optical intent is installed, the new link is discovered by PckC by means of periodically generated LLDP packets. Thus, PckC updates the parent topology view to HrC, using a REST POST method (message 994, label 4), and the time elapsed between message 977 and message 994

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

No.	Time	Source	Destination	Protocol	Info	
111	6.109920	HrC	OptC	HTTP	GET /onos/v1/devices HTTP/1.1	1
115	6.116423	OptC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
221	6.211098	HrC	OptC	HTTP	GET /onos/v1/devices/ports HTTP/1.1	
254	6.219084	OptC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
435	6.290393	HrC	OptC	HTTP	GET /onos/v1/links HTTP/1.1	
439	6.295229	OptC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
581	8.649063	HrC	PckC	HTTP	GET /onos/v1/devices HTTP/1.1	2
586	8.674913	PckC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
700	8.780077	HrC	PckC	HTTP	GET /onos/v1/devices/ports HTTP/1.1	
708	8.792550	PckC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
811	8.870277	HrC	PckC	HTTP	GET /onos/v1/links HTTP/1.1	
815	8.881243	PckC	HrC	HTTP/JSON	HTTP/1.1 200 OK	
938	11.324813	HrC	OptC	HTTP/JSON	POST /onos/optical/intents HTTP/1.1	3
941	11.330625	OptC	HrC	HTTP	HTTP/1.1 201 Created	
977	11.852513	OptC	HrC	HTTP/JSON	POST /onos/HrC-app/listener/opticalintent	4
994	12.956186	PckC	HrC	HTTP/JSON	POST /onos/HrC-app/listener/links	
1001	12.970606	HrC	PckC	HTTP/JSON	HTTP/1.1 202 Accepted	5
1012	12.992710	HrC	PckC	HTTP/JSON	POST /onos/v1/intents HTTP/1.1	
1015	13.012442	PckC	HrC	HTTP	HTTP/1.1 201 Created	
1018	13.012966	HrC	PckC	HTTP/JSON	POST /onos/v1/intents HTTP/1.1	6
1020	13.031411	PckC	HrC	HTTP	HTTP/1.1 201 Created	

Fig. 14. Packets capture at HrC during Shared workflow execution.

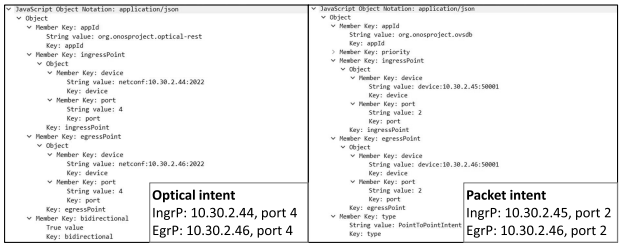

Fig. 15. Packet and optical intent configuration in the Shared workflow.

is equivalent to T_{link}. Then, HrC requests the configuration of the packet intent to PckC (messages from 1001 to 1020, labels 5 and 6). Due to implementation reasons, in this case, two separate requests are used to establish a pair of unidirectional packet intents, while in the Excl case, a single request is generated from Hrc to PckC asking for a bidirectional packet intent. Thus, the time elapsed between message 994 and message 1020 is equivalent to T_{pck}.

Looking at the capture time stamps, the procedure in this case takes around 2 s to perform the end-to-end intent configuration spanning across the packet and optical domains. In more detail, $T_{e2e} = 1.71$ s, $T_{\text{opt}} = 0.53$ s, $T_{\text{link}} = 1.10$ s, and $T_{\text{pck}} = 0.08$ s.

Figure 15 shows details related to the messages used by HrC to request an optical intent to OptC and a packet intent to PckC. On the left, the optical intent request message is shown, including the fields ingressPoint, egressPoint, and bidirectional. In this case, the ingress and egress points for the optical intent are the two pluggables installed in the packet–optical nodes (i.e., 10.30.2.44 port 4 and 10.30.2.46 port 4). On the right, the packet intent details are shown. In this case, the ingressPoint and the egressPoint are mapped to the packet ports of the packet–optical nodes where the traffic generator interfaces are connected (i.e., respectively, 10.30.2.44 port 2 and 10.30.2.46 port 2).

Figure 16 presents the time distribution needed to complete the overall workflow (i.e., T_{e2e}). Specifically, the experimental configuration of an end-to-end intent has been repeated 30 times, where each experiment encompasses steps 3, 4, 5, and 6 as reported in Fig. 14. The measured time includes the time that OptC needs to configure the optical parameters in both the ROADMs and pluggables located in the packet–optical

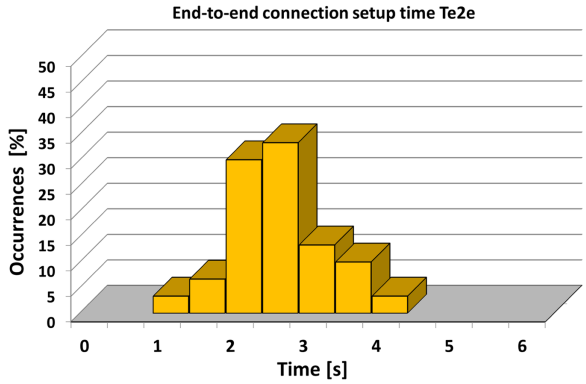

Fig. 16. Distribution of T_{e2e} over 30 experiments using the Shared workflow, average value $\bar{T}_{e2e} = 2.38$ s.

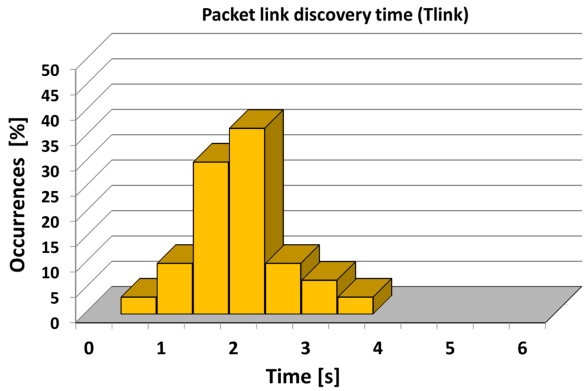

Fig. 17. Distribution of T_{link} over 30 experiments using the Shared workflow, average value $\bar{T}_{\text{link}} = 1.78$ s.

nodes. The average value registered for the end-to-end setup time is 2.38 s (i.e., $\bar{T}_{e2e} = 2.38$ s). In addition, the histogram shows that more than 80% of the samples are distributed in the range from 2 and 3 s, without samples above 4.5 s.

As happened for the Excl workflow, among the considered contributions, the experimental data revealed that both T_{opt} and T_{pck} are negligible. Indeed, the former is distributed in the range from 0.49 to 0.63 s (accounting for the time needed for the requested intent to arrive in the state INSTALLED), while the latter ranges from 0.04 to 0.37 s.

Figure 17 illustrates the distribution of the link discovery time T_{link} that has been demonstrated to be the most relevant contribution to T_{e2e}; i.e., on average, T_{link} counts for around 75% of the overall T_{e2e}. In this case, the measured time is the time elapsed at HrC between the reception of the optical intent installation event (generated at OptC) and the reception of the packet link discovery event (generated at PckC). In this case, the average value registered for the end-to-end setup time is 1.78 s (i.e., $\bar{T}_{\text{link}} = 1.78$ s). As shown in the histogram, around 75% of the samples are below 2.5 s, without samples above 4 s. As explained before, this result could be improved by increasing the generation rate of LLDP packets at PckC.

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

3. Performance Comparison

The distributions obtained for the end-to-end connection setup time T_{e2e} using the two considered workflows are illustrated in Figs. 12 and 16. The results prove that the Shar approach guarantees a faster T_{e2e}: the average value is faster by about 34% (3.59 s for Excl and 2.38 s for Shar). This happens because using the Excl approach, the configuration of the optical devices is performed by two different controllers (i.e., OptC configures ROADMs, and PckC configures pluggables). Thus, to allow the Excl workflow, additional information is required to be exchanged between the two child controllers (e.g., lightpath central frequency decided by OptC), and therefore, the configuration of pluggables is executed only when the configuration of ROADMs is fully completed. Conversely, with the Shar workflow, the configuration parameters of ROADMs and pluggables are decided at OptC, and their actual configuration is triggered in parallel, thus leading to a faster end-to-end connection setup.

On the functional side, both solutions are able to set up the connectivity. To evaluate the performance in terms of success rate on establishing connections, a careful simulation study could be useful. However, since the two solutions differ only in the order of the configuration actions but apply the same resource selection algorithms (e.g., the applied routing and spectrum assignment by OptC), the output in terms of the success rate is expected to be almost the same. The only difference could be due by the setup time; indeed, as demonstrated in [32], a shorter configuration time can lead to a slightly better success rate because the probability of running into resource contentions using a hierarchy of controllers is influenced by the configuration time.

6. CONCLUSIONS

This paper considered the emerging heterogeneous network scenario that combines the packet and optical domains, including the hybrid packet–optical nodes, e.g., packet routers equipped with coherent pluggable transceivers. Two hierarchical approaches, Excl and Shar, are proposed to coordinate and control such networks relying on a HrC and two child SDN controllers, each for one of the two underlying domains. The two approaches have been presented, including the related workflows and architectural implementation, with particular focus on the pluggables configuration and the communication among the three SDN controllers. Both approaches have been implemented, validated, and compared in an experimental environment.

Experimental results show that the Shar workflow is faster, mainly due to a simpler coordination between the two child controllers. More specifically, the Excl workflow requires the exchange of detailed information, which leads to the serialization of the configuration procedure of several involved optical devices (e.g., pluggables and ROADMs). Conversely, with the Shar workflow, the configuration of all the optical devices can be performed in parallel, finally leading to a faster activation of the optical lightpath. Experiments are performed in a network scenario where only the test intent is present; thus, explicit feedback on the scalability of the proposed solutions is not provided. However, exploiting the hierarchical structure, both solutions significantly improve the network scalability with respect to the adoption of a single controller for both layers. The scalability of the two solutions is expected to be similar, as both perform a set of operations in response to each new request. The Excl approach could perform slightly better because it is faster to process the single request.

Additional practical implications and considerations should be taken into account when comparing the two proposed solutions. For example, maintenance procedures appear to be more complex in the Shar case. Indeed, since packet–optical nodes are accessed by both controllers, before performing a NOS version update on the nodes, compatibility has to be guaranteed toward both controllers, which are typically provided by different vendors. Conversely, in the Excl case, only one controller (i.e., vendor) interacts with each type of node. Further considerations might refer to commercial/proprietary development of the controllers, which might include specific features that facilitate the management of one of the two solutions.

This paper lays the foundations for further research activities related to the management of packet–optical nodes equipped with coherent pluggable transceivers to support other transmission use cases (e.g., selection of an operational mode and modulation format based on physical optical impairments estimation) and the recovery of network failures.

Funding. This work was supported by the European Union's Horizon 2020 Framework Programme under grant agreement no. 101016663 (B5G-OPEN) and was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of the NextGenerationEU partnership on "Telecommunications of the Future" (PE00000001, program "RESTART").

REFERENCES

1. J. Pedro and S. Pato, "High-performance versus power-efficient coherent optical interfaces: spectral efficiency and hardware count comparison," in *22nd International Conference on Transparent Optical Networks (ICTON)* (IEEE, 2020).
2. A. Gumaste, M. Sosa, H. Bock, and P. Kandappan, "Optimized IP-over-WDM core networks using Zr+ and flexible muxponders for 400 Gb/s and beyond," J. Opt. Commun. Netw. **14**, 127–139 (2022).
3. F. Cugini, D. Scano, A. Giorgetti, A. Sgambelluri, L. De Marinis, P. Castoldi, and F. Paolucci, "Telemetry and AI-based security P4 applications for optical networks [Invited]," J. Opt. Commun. Netw. **15**, A1–A10 (2023).
4. A. Minakhmetov, T. Zami, B. Lavigne, and A. Ghazisaeidi, "ANN-based evaluation of FOADM impact on 400Zr+ channels in WDM ring networks," in *27th OptoElectronics and Communications Conference (OECC) and 2022 International Conference on Photonics in Switching and Computing (PSC)* (2022).
5. https://www.neophotonics.com/press-releases/?newsId=12486.
6. E. Pincemin, Y. Loussouarn, Y. Pan, G. Miller, A. Gibbemeyer, B. Mikkelsen, A. Gaibazzi, W. Way, T. Yamazaki, A. Hayashi, and K. Fujiyama, "Interoperable CFP-DCO and CFP2-DCO pluggable optic interfaces for 100G WDM transmission," in *Optical Fiber Communication Conference (OFC)* (2019), paper Th1I.3.
7. M. Newland, R. Schmogrow, M. Cantono, V. Vusirikala, and T. Hofmeister, "Open optical communication systems at a hyperscale operator," J. Opt. Commun. Netw. **12**, C50–C57 (2020).
8. https://news.windstream.com/news.
9. OIF Physical and Link Layer (PLL) Working Group, "Implementation Agreement for Coherent Module Management Interface Spec" (2020).

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.

10. A. Giorgetti, A. Sgambelluri, R. Casellas, R. Morro, A. Campanella, and P. Castoldi, "Control of open and disaggregated transport networks using the Open Network Operating System (ONOS) [Invited]," J. Opt. Commun. Netw. **12**, A171–A181 (2020).
11. A. Sgambelluri, A. Giorgetti, D. Scano, F. Cugini, and F. Paolucci, "OpenConfig and OpenROADM automation of operational modes in disaggregated optical networks," IEEE Access **8**, 190094 (2020).
12. V. Lopez, W. Ishida, A. Mayoral, T. Tanaka, O. Gonzalez de Dios, and J. P. Fernandez-Palacios, "Enabling fully programmable transponder white boxes [Invited]," J. Opt. Commun. Netw. **12**, A214–A223 (2020).
13. E. Riccardi, P. Gunning, O. Gonzales de Dios, M. Quagliotti, V. Lopez, and A. Lord, "An operator view on the introduction of white boxes into optical networks," J. Lightwave Technol. **36**, 3062–3072 (2018).
14. 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).
15. R. Casellas, R. Martínez, R. Vilalta, and R. Muñoz, "Abstraction and control of multi-domain disaggregated optical networks with OpenROADM device models," J. Lightwave Technol. **38**, 2606–2615 (2020).
16. C. Manso, R. Munoz, N. Yoshikane, R. Casellas, R. Vilalta, R. Martinez, T. Tsuritani, and I. Morita, "TAPI-enabled SDN control for partially disaggregated multi-domain (OLS) and multi-layer (WDM over SDM) optical networks [Invited]," J. Opt. Commun. Netw. **13**, A21–A33 (2021).
17. A. Mayoral, V. López, M. López-Bravo, D. García-Montes, O. Gonzalez-de Dios, A. Aguado, R. Szwedowski, K. Mrówka, F. Marques, Z. Stevkovski, D. Verchere, Q. Pham-Van, L. Tancevski, and J.-P. Fernandez-Palacios, "Multi-layer service provisioning over resilient software-defined partially disaggregated networks," J. Lightwave Technol. **38**, 546–552 (2020).
18. D. Scano, A. Giorgetti, A. Sgambelluri, E. Riccardi, R. Morro, F. Paolucci, P. Castoldi, and F. Cugini, "Hierarchical control of SONiC-based packet-optical nodes encompassing coherent pluggable modules," in *European Conference on Optical Communication (ECOC)* (2021).
19. A. Sgambelluri, D. Scano, A. Giorgetti, F. Paolucci, E. Riccardi, R. Morro, P. Castoldi, and F. Cugini, "Coordinating pluggable transceiver control in SONiC-based disaggregated packet-optical networks," in *Optical Fiber Communication Conference (OFC)* (2021), paper W1G.3.
20. O. Gerstel, G. Galimberti, and B. Foster, "A control hierarchy for integrated packet-optical networks utilizing pluggable transceivers," in *Optical Fiber Communication Conference (OFC)* (2022), paper M4F.2.
21. R. Vilalta, R. Casellas, L. Gifre, R. Muñoz, R. Martínez, A. Pastor, D. López, and J. Fernández-Palacios, "Architecture to deploy and operate a digital twin optical network," in *Optical Fiber Communication Conference (OFC)* (2022), paper W1F.4.
22. The P4 Language Consortium, "$P4_{16}$ Language Specification," 2022, https://p4.org/p4-spec/docs/P4-16-v1.2.3.pdf.
23. The P4.org API Working Group, "P4Runtime Specification," 2022, https://p4.org/p4-spec/p4runtime/v1.3.0/P4Runtime-Spec.pdf.
24. L. Velasco, S. Barzegar, F. Tabatabaeimehr, and M. Ruiz, "Intent-based networking and its application to optical networks [Invited Tutorial]," J. Opt. Commun. Netw. **14**, A11–A22 (2022).
25. Sonic, http://sonic-net.github.io/SONiC/.
26. "Open Network Install Environment," 2022, https://github.com/opencomputeproject/onie.
27. "PINS," 2022, http://opennetworking.org/pins/.
28. "Stratum," 2022, http://opennetworking.org/stratum/.
29. "Open Network Linux," http://opennetlinux.org/.
30. "Bmv2," 2022, http://github.com/p4lang/behavioral-model.
31. "ONOS," https://opennetworking.org/onos/.
32. A. Giorgetti, "Proactive H-PCE architecture with BGP-LS update for multidomain elastic optical networks [Invited]," J. Opt. Commun. Netw. **7**, B1–B9 (2015).

Authorized licensed use limited to: Scuola Superio Sant'Anna di Pisa. Downloaded on August 02,2023 at 20:17:46 UTC from IEEE Xplore. Restrictions apply.