SDN-enabled OPS with QoS guarantee for reconfigurable virtual data center networks

SDN-enabled OPS with QoS Guarantee for Reconfigurable Virtual Data Center Networks

Wang Miao, Fernando Agraz, Shuping Peng, Salvatore Spadaro, Giacomo Bernini, Jordi Perelló, Georgios Zervas, Reza Nejabati, Nicola Ciulli, Dimitra Simeonidou, Harm Dorren and Nicola Calabretta

***Abstract*—Optical packet switching (OPS) can enhance the performance of data center networks (DCNs) by providing fast and large-capacity switching capability. Benefiting from the Software-Defined Networking (SDN) control plane which could update the look-up-table (LUT) of the OPS, virtual DCNs can be flexibly created and reconfigured. In this work, we have implemented and assessed an SDN-based control framework for an OPS node, where the OpenFlow protocol has been extended in support of the OPS switching paradigm. Application flows are switched by the OPS at sub-microsecond hardware speed, decoupled from the slower (milliseconds timescale) SDN control operation. By the DCN infrastructure provider, the virtual networks become directly programmable with the abstraction of the underlying OPS node. Experimental results validate the successful setup of virtual network slices for intra-data center interconnect and quality of service (QoS) guarantee for high-priority application flows. Data plane resources are efficiently shared by exploiting statistical multiplexing. In addition, the capability of exposing per-port OPS traffic statistics information to the SDN controller enables the implementation and experimental validation of load balancing algorithms to improve the QoS performance.**

***Index Terms*—Optical interconnects; Optical packet switching; Software defined networking;**

I. Introduction

Today, data centers (DCs) are experiencing the rapid development of information and communication technology (ICT) markets, where a broad range of emerging services and applications are offered [1-3]. Multi-tenancy enabling efficient resource utilization is considered as a key requirement for the next generation DCs resulting from the growing demands for services and applications. Therefore virtualization mechanisms and technologies are deeply implemented by DC providers to efficiently multiplex customers within their physical network and IT infrastructures [4,5]. The end users and business customers do not need to maintain their own physical IT infrastructures, while offering a scalable, simple to provision and cost-effective virtual solution for computing, storage and integrated applications [6,7]. In this scenario, a key innovation is the creation of multiple independent Virtual Networks (VNs) which logically ensure the interconnectivity. Dedicated routing policy can be applied for different tenants based on abstracted switching resources, like traffic flows. Data Centre Network (DCN) virtualization therefore enables additional benefits for service providers and enterprises, improved resource utilization, rapid service delivery, flexibility and mobility of applications within DCs [8,9]. Moreover, network virtualization mechanisms can leverage statistical multiplexing and fast switch reconfiguration to further extend the DC efficiency and agility.

Current DCNs are organized in a multi-tier topology where top-of-rack (ToR) switches interconnect groups of servers and the ToRs are in turn optically interconnected by electronic switches [10]. This architecture supports statistical multiplexing and allows for hardware sharing at server level (e.g. virtual machine (VM)) providing operational efficiency. However, there are hardware and control issues that limit the application of this model for next-generation DCNs. Firstly, the total I/O bandwidth of the switch IC is limited by the size of ball grid array (BGA) at the package [11]. Moreover the tree-like topology has intrinsic scaling issues in terms of bandwidth and latency [12]. In addition to the hardware infrastructure, the provisioning of VNs requires a proper abstraction first, and configuration later, of the DCN resources [8]. The proprietary control interfaces of the switches are no longer efficient or effective for next paradigm DCNs. On the contrary, a standardized solution could help delivering significant levels of agility, speed, and manageability [13].

To overcome the hardware limitations of electrical DCNs, optical switching technologies exploiting space, time, and wavelength multiplexing have been investigated in several projects for effectively accommodating DC applications [14]. Optical Circuit Switching (OCS) based solutions have been

Manuscript received February 25, 2015.

W. Miao is with the Electrical Engineering Department, Eindhoven University of Technology, Eindhoven 5600 MB, The Netherlands (e-mail: w.miao@tue.nl).

presented in [15-17], where high-bandwidth connections can be provided although the slow reconfiguration time limits the flexibility and operations for latency-sensitive applications. Alternative solutions relying on Optical Packet Switching (OPS) allow for fine (sub-wavelength) granularity thus significantly improving the bandwidth efficiency and additionally providing fast switching capability [18,19]. Although potential scalability to large port-count has been investigated, OPS with sub-microsecond reconfiguration time has only been demonstrated with limited radix mainly due to the complexity and the lack of optical buffer. To effectively cope with both long-lived and short-lived traffic flows, a novel flat intra-DCN architecture LIGHTNESS [20] integrating OCS and a scalable wavelength, space, and time switching OPS with optical flow control for solving packet contentions [21] was introduced. Software Defined Networking (SDN) has been chosen as the base control technology to facilitate network provisioning and virtualization. By fully abstracting the underlying data plane devices, the SDN controller can effectively create and manage multiple VNs. In LIGHTNESS, an SDN controller for OCS node has been developed and assessed [22], while the OPS is currently controlled by a proprietary interface not compliant with the SDN controller [23]. To date, OpenFlow (OF) extensions for a different OPS architecture based only on time and space switching have been presented in [24]. However, novel OF extensions for managing all the wavelength, space, and time switching elements of the LIGHTNESS OPS node has to be developed and assessed. In addition, specific OF implementation and extensions are also needed to manage the optical flow-control for solving contentions in LIGHTNESS OPS.

In this paper we present and experimentally demonstrate an SDN-enabled OPS node for reconfigurable DCNs. OF protocol has been extended for managing all the wavelength, space, and time switching elements and the flow-controlled OPS node to fully support SDN-based control. An OF agent is implemented to facilitate the communication with control plane through the southbound interface (SBI). Based on this, the LUT of the OPS could be dynamically updated by the SDN controller to facilitate the virtualization and management of the network. Once the virtual DCN is provisioned, the flow-controlled OPS node provides sub-microsecond hardware switching, decoupled from the milliseconds of SDN operation time. In addition, benefiting from the flow control operation, the traffic statistics are collected by the control plane which enables network optimization functions including priority assignment and load balancing. Thus the network can be dynamically controlled and operated resulting in significant system flexibility and controllability. The developed SDN-based control functions for the OPS VNs are experimentally assessed. Results show successful VN reconfiguration and QoS guaranteed operation for high-priority application flows by manipulating the LUT. Exploiting the monitoring capability of the real time traffic statistics, a load balancing algorithm for network performance optimization is implemented and assessed.

The paper is organized as follows. Section II describes the system under investigation including both control plane and OPS node. Section III presents the specific control plane function as well as the OF extensions implemented to enable the communications between the SDN controller and the OPS OF agent. The validation and assessment of the VN generation and reconfiguration, QoS guarantee via priority assignment, and load balancing based on the statistics collection are reported in section IV. Finally, section V concludes the paper by discussing the main results.

II. LIGHTNESS DCN ARCHITECTURE

The overall architecture of the SDN-enabled and programmable optical DCN is shown in Fig. 1 [20]. Therein, a cluster of racks is interconnected by an Architecture-on-Demand (AoD) node which consists of an optical backplane, i.e. a large port-count fiber switch. The role of the AoD is two-fold: first it provides OCS connectivity between ToRs, and second it provides a flexible optical infrastructure to interconnect with OPS and inter-cluster AoD. In each rack, a ToR switch groups tens of servers. Two switching elements, the OCS and the OPS are in charge of switching the intra-cluster traffic generated by the VMs. The ToR aggregates the traffic into flows and sends them to the proper switching element (i.e., the OCS or the OPS) based on the traffic type or flow size. Hence, the OCS handles the long-lived rack-to-rack flows and the OPS processes the short-lived bursty ones. Similar to intra-cluster communication, the inter-cluster communication can be dynamically configured with an inter-cluster backplane.

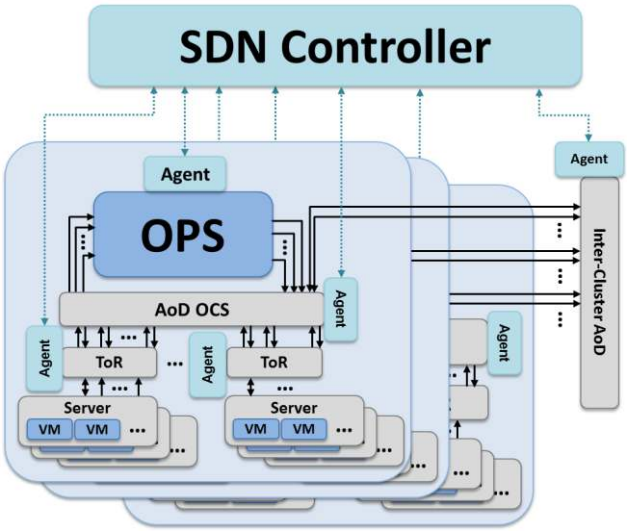

Fig. 1 Overall architecture of LIGHTNESS DCN

The work presented here focuses on the performance assessment of OPS-based VNs enabled by the SDN control. The fast reconfiguration time of OPS node provides high flexibility and utilization to the VNs, compared with OCS, which needs much longer reconfiguration time [14]. Triggered by the DC operator or an external application leveraging the programmability exposed at the northbound interface, the SDN controller can dynamically configure OPS-based VNs over this scenario, following a top-down approach. The OPS-based VN is created and managed by the DC infrastructure provider and it consists of a collection of flows associated to a single tenant. In turn, a flow is defined

here as a set of application data packets that are aggregated into optical packets containing the same optical label with a certain load. As will be further explained, the load and the priority of the flows can be manipulated to provide guaranteed QoS in terms of packet loss and latency. The tenant is allowed to run applications on its own VN and makes changes to it when given the authority.

A. SDN-based Control Plane

This paper proposes a decoupled control plane deployed on top of the flat optical DCN to support network re-configurability and programmability. It consists of a centralized SDN controller interacting with the DC network devices through the SBI, which implements an extended version of the OpenFlow (OF) protocol. A dedicated OF agent is deployed on top of each switching device as a mediation entity between the SDN controller and the proprietary control interfaces exposed by the devices. Each agent provides an abstraction of the corresponding DCN device and enables a uniform resource description at the SDN controller level. The SDN controller translates the requirements generated from the application plane and accordingly configures the data plane devices. Facilitated by the SDN control and related protocols, once the virtual data center infrastructure is provisioned, the traffic flows generated by its owner's applications are automatically classified, recognized and associated to the given VN. Logical isolation (e.g. VLANs) can be performed to avoid the interference among the traffic carried in different VNs.

To set up an OPS-based VN, the SDN controller configures the look-up tables (LUTs) of the ToRs for those racks whose servers have to be interconnected. Moreover, the controller configures the LUT of the OPS to properly interconnect those ToRs. Figure 2 presents an example of VN creation and reconfiguration in an OPS-enabled DCN. *VN1* connects *ToR1* with *ToR2* while *VN2* interconnects *ToR2*, *ToR3* and *ToRN*, with *ToR2* belonging to both VNs. Here, we assume that the tenant owning *VN1* intends to run a new application flow in a VM hosted in *Rack3*. In this case, *ToR3*, which connects *Rack3* to the DCN, has to be included in *VN1* so a network

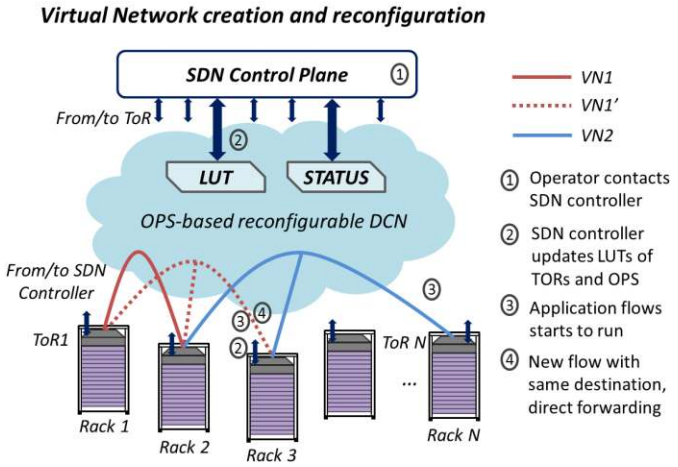

Fig. 2 Virtual Network creation and reconfiguration

reconfiguration is required. As said a top-down approach is used here. This means that the operation is triggered by the DC management by means of the SDN controller, which in turn updates the LUTs of the ToRs (*ToR1*, *ToR2* and *ToR3*) and the OPS involved in the new *VN1* (*VN1'*). Once the VN has been provisioned, application data are exchanged between the ToRs, thus decoupling the data plane (at sub-microsecond time scale) from the SDN controller (at milliseconds time scale). Moreover, exploiting the statistical multiplexing introduced by the OPS, the bandwidth/wavelength resource sharing is possible between flows associated either to the same VN or to different VNs. This enables the dynamic creation and reconfiguration of multiple VNs and the optimization of the DCN resources utilization, which leads to a high tenant density.

B. OPS Data Plane

The data plane scheme of the implemented OPS node is presented in Fig.3. Benefiting from the modular structure, the switch exploits a highly distributed control so that reconfiguration time (~ns) is constant regardless of the port-count. Moreover, scaling to large number of ports can be simply realized by deploying copies of the single module [21].

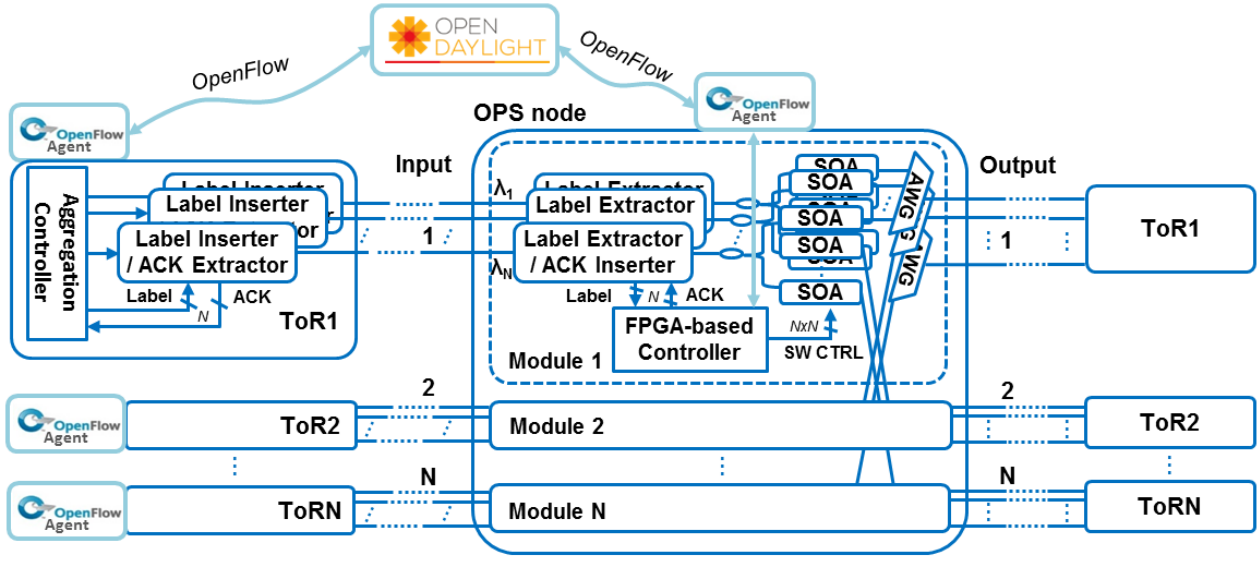

Fig. 3 SDN-enabled OPS

The optical flow generated and transmitted by the ToR includes an optical label that, according to the OPS LUT stored in an FPGA-based switch controller, determines the forwarding output port at the OPS. The switch controller manages the SOA gates to forward the packets to the desired destinations. Due to the statistical multiplexing, contention may happen between the input signals from the same ToR. Therefore, optical flow control signals notifying the packets delivery status are implemented between the ToRs and the OPS node. The FPGA-based OPS switch controller generates an ACK signal, which is sent back to the ToR, to notify a successfully delivered packet. Otherwise, a negative ACK (NACK) is generated for a blocked packet and, then, the packet has to be retransmitted [25]. Such bi-directional optical flow control operation brings sub-microsecond average end-to-end latency and 10^{-5} packet loss for a normal traffic load (<0.5) [21]. Apart from the forwarding operation, the FPGA-based switch controller also records the numbers of received packets and NACKs (contentions). These values will be reported to the SDN controller and can be reset to zero once the flow transportation has ended.

III. OpenFlow Extensions for the OPS Node

Figure 4 depicts the control architecture deployed for the scenario under investigation. The SDN controller is implemented by means of the OpenDaylight (ODL) platform, an open source initiative hosted by the Linux Foundation in its Hydrogen release (Base edition) [26]. ODL has an extensible modular architecture as well as a wide set of available services, appliances and northbound control primitives suitable for the SDN-enabled OPS described in this paper. However, the ODL controller requires a set of extensions in support of the OPS technology. First, the OF protocol driver at the SBI side of the ODL controller was modified by applying a set of OPS extensions defined for the OF protocol. This enabled the control and provisioning of OPS devices from the controller. Afterwards, a set of core services of the ODL controller including the Forwarding Rules Manager, the Service Abstraction Layer and the Topology Manager (responsible for managing the topology) were also extended in support of OPS devices and flows. Finally, the ODL management Graphical User Interface (GUI) was extended to allow the management of OPS-capable devices as well as to configure and manage OPS-based flows.

Besides this, the ToRs and the OPS node were equipped with OF agents that enable the communication with the ODL controller through the SBI, therefore bridging and gluing control and monitoring mechanisms and primitives at both sides. More specifically, the OF agents, which run in dedicated servers, map the proprietary interface exposed by the OPS switch controller and aggregation controller in the ToR into the extended version of the OF protocol implemented at the ODL SBI (Fig. 4). In this way, the agents translate the OF messages coming from the controller into a set of actions performed over the underlying device through the proprietary interface and vice versa. Hence, the SDN controller is able to configure the OPS DCN flexibly by updating the LUT, while on the other hand, the status of the underlying hardware switches is reported up to the SDN

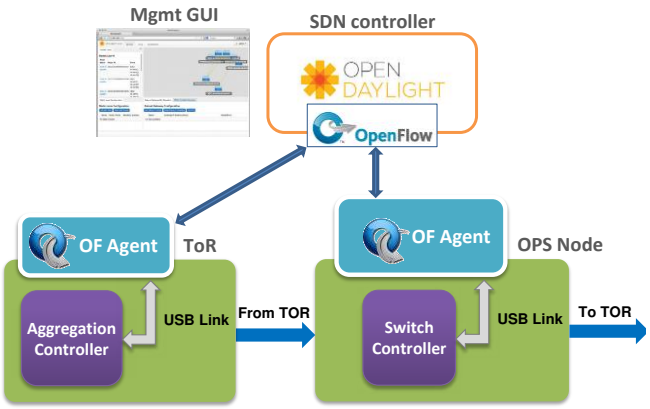

Fig. 4 Control architecture

controller for monitoring purposes. The ODL management GUI allows for triggering and visualizing these actions.

The OF protocol [27] has arisen as a *de facto* standard to implement the SBI defined by the SDN paradigm. OF allows moving the network control out of the devices to the control plane, since it enables the manipulation of the forwarding tables of such network devices. In this way, data flows can be automatically configured to satisfy users' needs dynamically. To this end, the OF protocol defines a set of messages and attributes, which are exchanged between the controller and the network elements, to configure the data flows. The most relevant messages of the protocol to this work are the FEATURES_REQUEST/REPLY message pair, the FLOW_MOD and the STATS_REQUEST/REPLY pair. The FEATURES_REQUEST message is sent by the controller to the network device to request the capabilities (i.e. switching technology, number of ports, etc.) of the device. The network element sends back a FEATURES_REPLY to satisfy the controller's request. The controller uses the FLOW_MOD message to configure, modify or delete data flows. To this end, the operation (i.e. *New Flow*, *Flow Modify* or *Delete Flow*) and the characteristics of the flow are conveyed in the message. Finally, the controller requests statistics from the data plane by means of the STATS_REQUEST message. Different types of statistics (port, table, flow, …) can be requested. Upon the reception of these requests, the network devices send back the associated STATS_REPLY messages containing the required values.

However, the current OF specification focuses on EPS devices, so some modifications are needed to enable OF in the proposed optical data plane. Different from the previous work presented in [23], where a proprietary control protocol was used to configure the optical network devices, for this paper the OF v1.0 protocol was extended to fully support OPS technology and its specific switching paradigm. In particular, the OPS switching feature was added to the *ofp_capability* attribute, which is conveyed in the FEATURES_REPLY message, so that the controller can recognize devices implementing this new switching technology. The *ofp_match* and *ofp_action* fields were extended as well to enable the OPS flow configuration by means of the FLOW_MOD message. More specifically, the OPS label and wavelength attributes extend the *ofp_match*, and the load of the OPS flow can be set thanks to a new action added to the *ofp_action* field. These extensions aim to

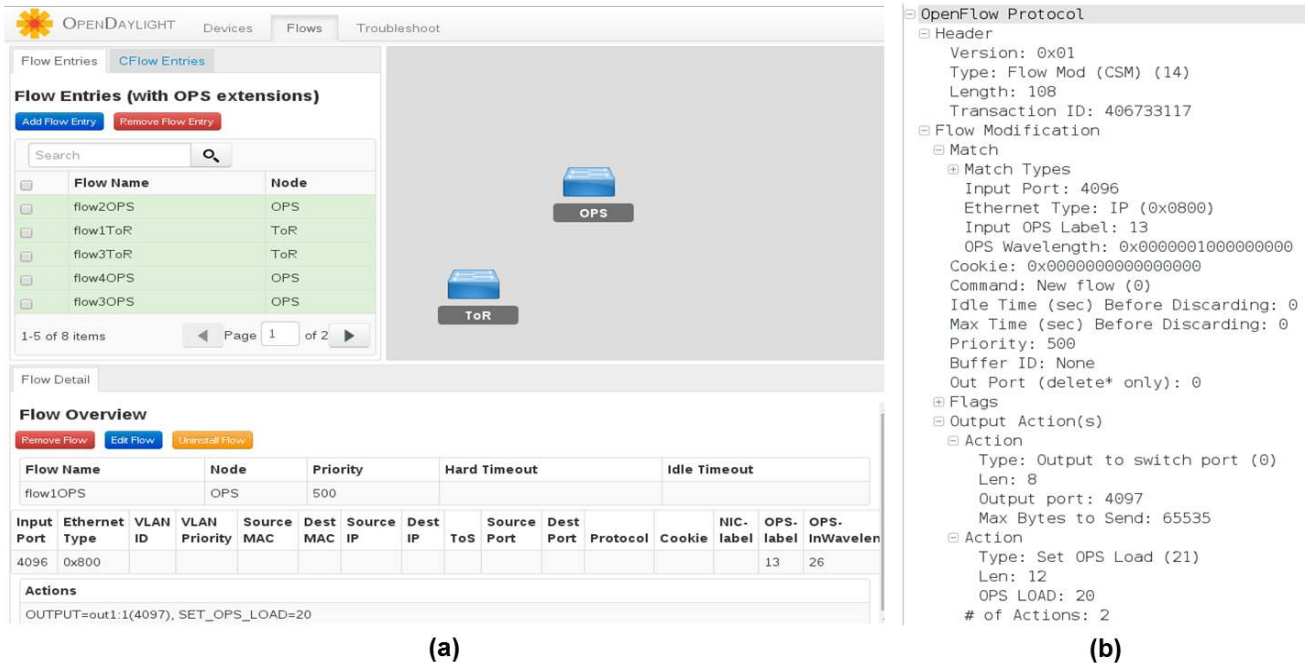

Fig. 5 (a) OpenDaylight GUI with OPS extensions; (b) OpenFlow FLOW_MOD extended message

support the configuration of the LUTs in both the OPS node and the ToR. During operation, the ToR uses the information stored in the LUT to configure the labels of the optical packets, and the OPS node uses it to switch the incoming packets to the appropriate output port. Figure 5(a) depicts the extended ODL management GUI. Concretely, it shows the ToR and the OPS node that have been detected by the controller with the extended FEATURES_REPLY message. In addition, Fig. 5(b) depicts an example of an extended FLOW_MOD message received by the OPS node to configure a new flow. Specifically, the figure presents the extended OF match field conveying the OPS label (13) and the wavelength bitmap. It is worth noting here that the wavelength bitmap follows the same format as the one proposed in the OCS extensions addendum for the OF protocol v1.0 [28]. The new OF action, which allows for setting the load (20) to the OPS flow, is also shown in the figure.

The standard STATS_REQUEST/REPLY message pair is used for the collection of statistics, since it already copes with the needs of the scenario under study. It is also worth noting here that only port statistics are considered in this work.

IV. Experimental Evaluation

Facilitated by the implemented OF agent and extended OF protocol, the SDN-based control functions are enabled for the OPS node. VNs can be created and managed remotely through the SDN control plane. To validate the VN flexibility, agility and QoS guarantee benefited from this, both data plane and control plane operations including VN reconfiguration, priority assignment and load balancing based on statistics collection have been experimentally investigated.

As illustrated in Fig. 3, an FPGA-emulated ToR performs the statistical multiplexing of the packets associated to the traffic flows and transmits them to the OPS node that, in turn, forwards the packets to the proper destination according to the attached labels. The FPGA-based switch controllers are interfaced to the OF agents through USB connections. For each packet, a 4-bit label contains the forwarding information (2 bits) and the class of priority (2 bits), and it is assigned by the ODL controller according to the application requirements. The ToR is equipped with an aggregation controller, which is responsible for aggregating the traffic coming from the servers of the rack, generating the optical packets and assigning the appropriate label to them (i.e. flow generation process). Furthermore, it implements the flow control mechanism at the ToR side. In particular, the buffer manager inside the aggregation controller stores the label information and performs the packet (re-)transmission according to the ACK/NACK sent by the OPS node. The gates used for controlling the transmission of packetized 40Gb/s NRZ-OOK payloads (460ns duration and 40ns guard time) are triggered by the buffer manager in case of (re-)transmission.

A. VN Reconfiguration

As said, the aggregation controller at the ToR side allocates a certain label for the incoming packet by matching the destination requirements with the LUT. Upon the reception of the packet, the OPS processes the label and forwards it to the corresponding output port according to the information provided by the LUT. However, as the demands of users and applications change, the created VNs need to be flexibly reconfigured and adapted to the dynamic requirements of the applications. In this case, the LUTs of both the ToR and the OPS nodes can be updated by means of the SDN controller to reconfigure the interconnection of the VNs according to the new requirements.

In the example depicted in Fig. 6, the ODL controller has originally provisioned *VN1*. Application flows, *Flow 1* and *Flow 2*, are statistically multiplexed on the same wavelength λ_1 and switched to different output ports. Different priority

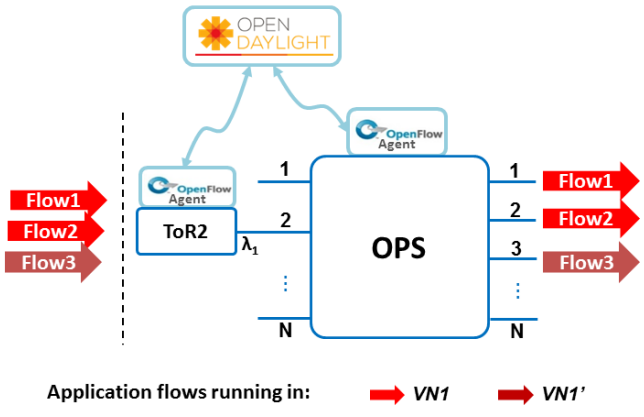

Fig. 6 Virtual networks reconfiguration

levels are assigned to each flow in case of potential resource competition between the flows. To support a newly generated *Flow 3*, a reconfiguration of *VN1* is required to provision the connectivity with output *Port 3*. To this aim, the DC management uses the ODL controller to update the LUTs in the ToR and the OPS through the OF interfaces exposed by the agents.

In this procedure, an OF FLOW_MOD message is sent to the OF-agents, which process the command. The message specifies the input and output OPS ports, and the proper label including the class of priority for the corresponding packets. Then the agents execute the configuration instructions for the ToRs and the OPS to update their LUTs so that one more LUT entry will be added. At this point, the VN has been reconfigured, and the OPS node supports the delivery for the new flow. Additionally, the ODL controller can be also used to disable a certain flow or to make modifications (such as adjusting priority) by deleting or editing the entries of the LUT, respectively. Figure 5(b) illustrates a detail of the OF FLOW_MOD message conveying a *New Flow* command. A summary of created flows listed on the GUI is given in Fig. 5(a).

In the data plane, Fig. 7 shows the LUT update for the original *VN1* (LUT) and the reconfigured *VN1'* (LUT'). There, "xx (L_4L_3)" represent the priority that, in case of collision, will be referenced to classify the priority in the order of "11>10>01>00". Note that, within *VN1*, there are no entry routes to the output *Port 3* in the LUT. The time traces of label L_2L_1 (L_4L_3 omitted), the incoming packets to the OPS (Flow in) marked with destination and outputs for the three ports are also plotted. The figure clearly shows that, before the reconfiguration (left side), flows destined to *ToR3* are dropped since no matching label is found in the LUT of the OPS node. On the contrary, once *VN1* is reconfigured and the LUT is updated (right side), the flows towards *ToR3* are then properly delivered with the L_2L_1 labeled to "11". The update process, which includes the communication between the ODL controller and both the ToR and the OPS, takes around 110ms; after that, flows are statistically multiplexed and switched. It is worth to note that the reconfiguration process does not affect other flows, so packets destined to *ToR1* and *ToR2* perform hitless switching during the VN reconfiguration time.

B. *Priority Assignment*

With statistical multiplexing, OPS-based VNs allow efficient resource sharing thus achieving high tenant density. However, as the traffic load increases, the competition for the physical resources may result into contention at the OPS node. The flow control mechanism introduced in section II aims at avoiding the data loss associated to contention. Nonetheless, this mechanism deteriorates the end-to-end latency performance due to the retransmissions and, once the buffer at the ToR side is fully occupied, the new coming packets will be lost. By assigning class of priority to data flows, the ones with higher priority will be directly forwarded without any retransmission. Following the top-down approach, the DC operator triggers the assignment of priority to a flow through the ODL controller. The extended OF enables this feature since the label information can be carried within the OF FLOW_MOD to configure the data plane. The label bits L_4L_3 define four different priority classes and the contention between the packets with the same priority is resolved here by means of round-robin scheduling.

As illustrated in Fig. 8(a), *Flow 4* and *Flow 5* are heading to the same output port (*Port3*) on different wavelengths. As they come from the same ToR and reach the same module of the OPS, there is a contention happening, and thus the priority class determines which packets are delivered and

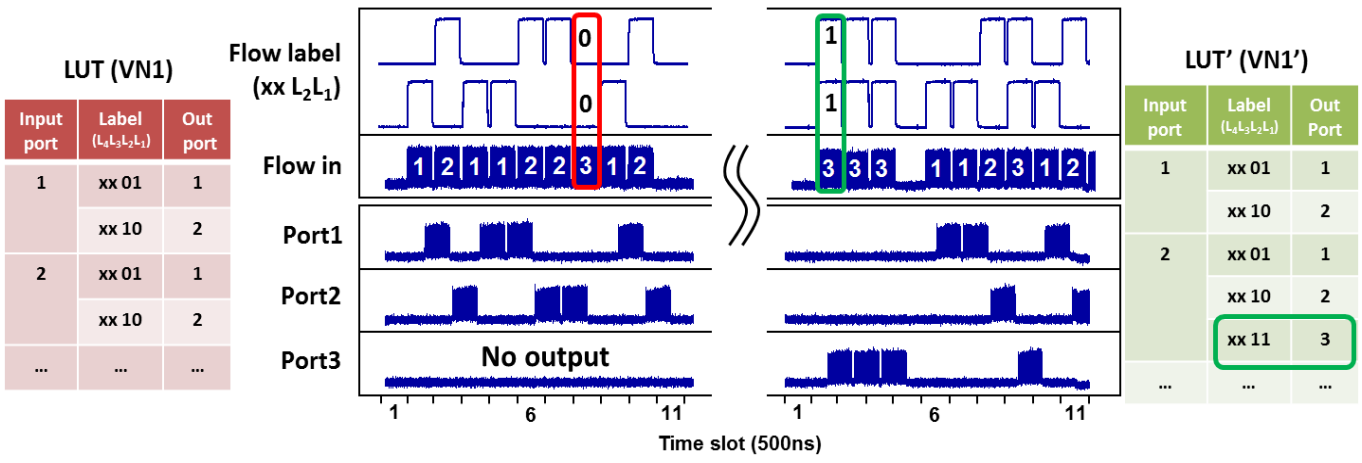

Fig. 7 Time traces for labels and packets before/after LUT update (VN reconfiguration)

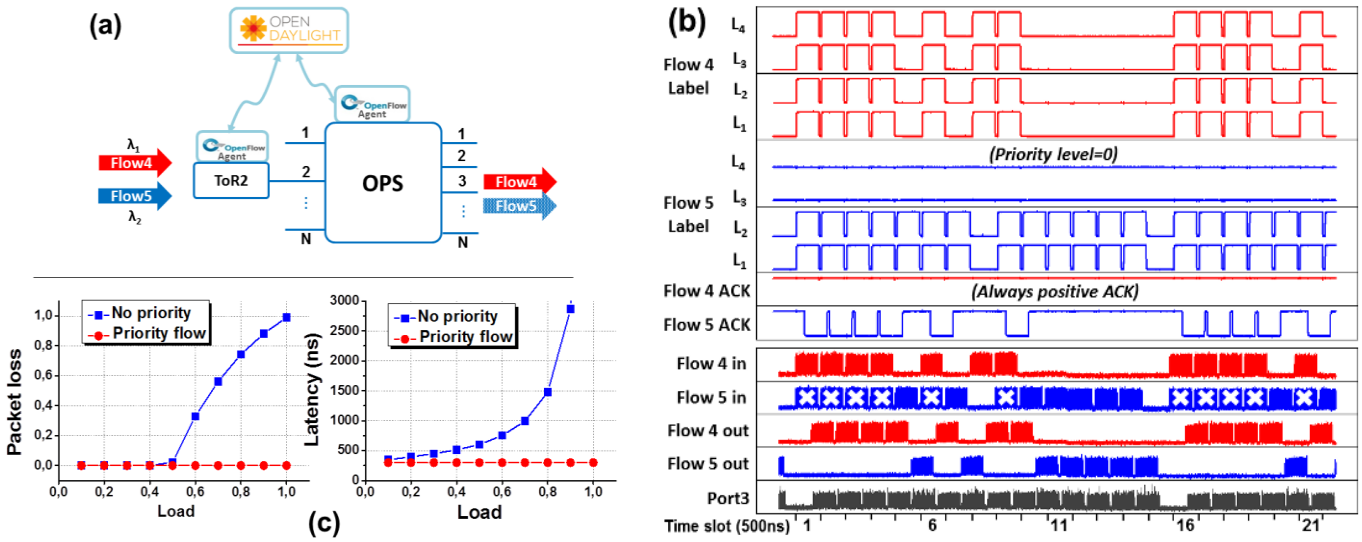

Fig. 8 (a) Priority assignment; (b) time traces of Flow 4 and Flow 5; (c) packet loss and latency

which ones are retransmitted. *Flow 4* has been assigned a higher priority (L_4L_3 = "11") than *Flow 5* (L_4L_3 = "00"). Therefore, in case of contention at the OPS, packets associated to *Flow 4* will be forwarded to the output *Port3* to avoid packet loss and higher latency caused by retransmission, while the ones associated to *Flow 5* will be blocked and then retransmitted. Figure 8(b) shows the label bits ($L_4L_3L_2L_1$), the flow control signals (ACKs), and the switching results for the two contented flows. The ACK signals for *Flow 4* are always positive (always forwarded) which means that all the packets are successfully delivered. *Flow 5* packets labeled with "x" are blocked due to the contention and a corresponding NACK is generated to ask for the retransmission. Figure 8(c) shows the packet loss and latency for both flows with a uniformly distributed load. The packet loss curves confirm no packet loss for *Flow 4*, while the 16-packet buffer employed at the ToR side prevents packet loss up to a load of 0.4 for *Flow 5*. For higher values of the load, as the buffer starts to be fully occupied, the packet loss increases linearly. The retransmissions observed for the blocked packets of *Flow 5* lead to an exponential increase of the latency. On the contrary, the priority assignment guarantees a low latency, and thus a high QoS for *Flow 4*.

C. *Statistics Report and Load Balancing*

As the centralized controller of the whole DCN, ODL is also in charge of monitoring the status of all the underlying devices. Based on the collected real-time information, the ODL controller provisions dynamic VNs updates and adjustments with the aim to improve the DC network efficiency and utilization. To this purpose, OF STATS_REQUEST/REPLY message pairs are exchanged between the ODL controller and the data plane devices, where the OF STATS_REPLY messages contain the statistical information provided by the optical devices. In particular, the OPS and ToR nodes collect the amount of processed data (in Kbytes) for both received and forwarded packets. The number of retransmissions due to the contention, which is essentially the number of NACKs, is also reported to the agent and included in the collisions field of the OF STATS_REPLY message.

Figure 9 illustrates the statistics collection messages for both the OPS (a) and the ToR (b). The example depicts the scenario described in the previous sub-section where two active flows face contention. The OPS switch controller records the number of received packets as well as the NACK signals for each flow. Once receiving a request for gathering the port statistics, the OF-agent reads the counters from its controlled device and reports the aggregated per-port values to the ODL controller through the OF STATS_REPLY message. Hence, the counters are translated into received and transmitted packets, and collisions (i.e. NACKs). Figure 9 (a) presents the detail of the OF STATS_REPLY message carrying the OPS ports statistics. This information is then depicted in the ODL GUI.

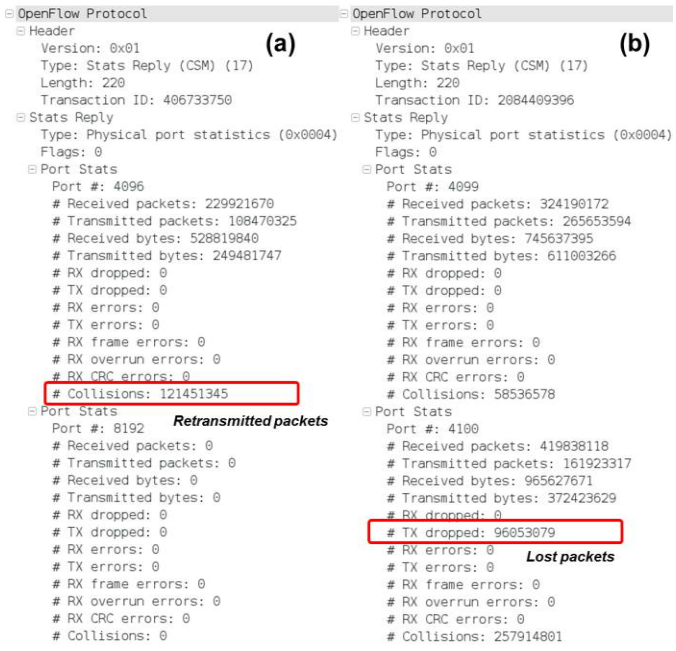

Fig. 9 OF STATS_REPLY messages for (a) OPS and (b) ToR

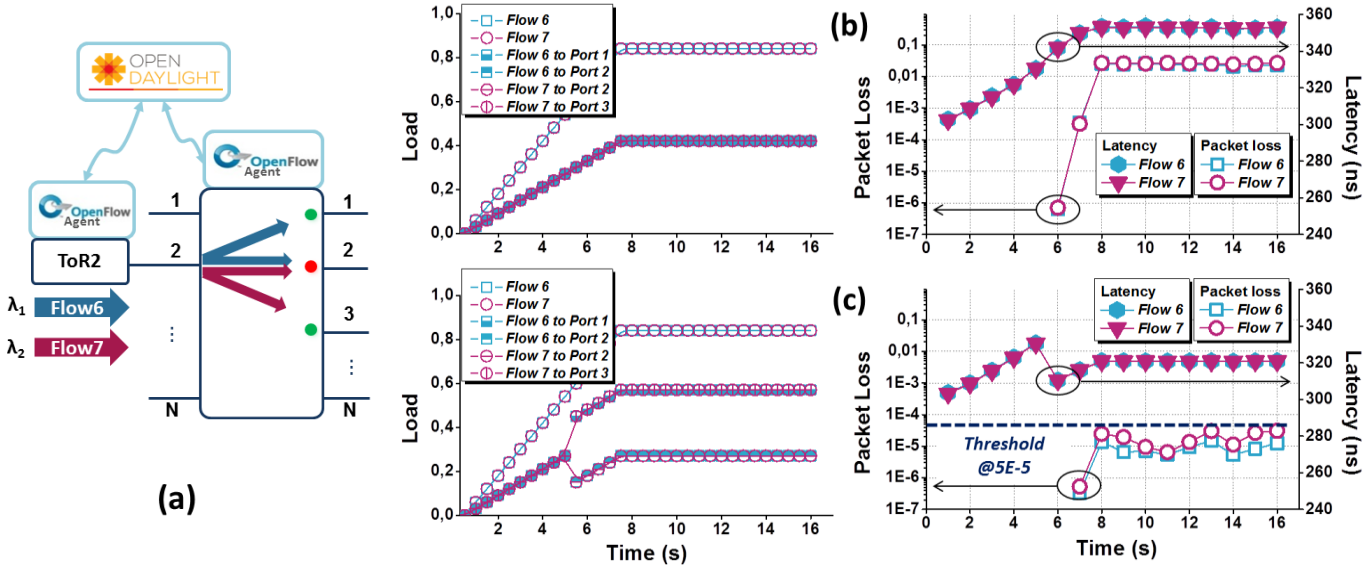

Fig. 10 (a) Load balancing operation; packet loss and latency changes without adjusting (b) and with balancing step of 0.15 (c)

For evaluation purpose, the packet loss, which affects the QoS significantly, is a parameter that needs to be tracked. Since the buffer is implemented in the ToR side, the packet loss performance can only be collected and reported to ODL from the ToR OF-agent. This has been implemented by utilizing the *TX dropped* field of the OF STATS_REPLY message as shown in Fig. 9 (b). The ODL controller can then be used to optimize the system performance according to the application requirements based on the statistic information reported by the OF-agents.

An example showing the load balancing operation based on statistics collection and flow modification is given in Fig. 10(a). Two flows belonging to two different VNs have common output *Port2*. As the load increases, the contention at *Port2* would cause high packet loss for both flows. Upon reception of the real-time status of the per-port packet loss and the occupancy of each alternative port from the ToR, the ODL can balance the load to the ports with less usage. As can be seen in Fig. 10(b), the load of both flows has been increased from 0 to 0.8, with 50% probability destining at *Port 2* at the beginning. If ODL does not update any of the LUTs, high packet loss is observed. In comparison, targeting a packet loss threshold of 5E-5, once the reported statistics tend to exceed this value, the load at *Port2* will be balanced to *Port1* (for *Flow6*) and Port3 (for *Flow7*) through the *Flow Modify* command. In this case, the adjustment is proactive when the detected retransmission rate (contention possibility) is higher than 10%. A balancing step of 0.15 has been set to properly avoid possible performance degradation with the given load increasing speed. According to the QoS settings, the packet loss <5E-5 and latency <340ns are guaranteed as shown in Fig. 10(c).

V. Conclusion

An SDN-enabled OPS for reconfigurable virtual DCNs has been investigated in this work. On one hand, the flow-controlled OPS with distributed control allows for sub-microsecond latency switching and large connectivity enabled by statistical multiplexing. On the other hand, the SDN-enabled VNs can be flexibly reconfigured and managed significantly improving the DC agility and controllability. The deployment of the SDN controller decouples the control plane from the underlying data plane so that the decisions are made based on the functional abstractions of the OPS without interfering the fast forwarding. The OpenFlow protocol has been chosen and properly extended to provision and dynamically update the VNs, and an OpenFlow agent has been implemented to facilitate the communication between the SDN controller and the OPS node.

The experimental assessment demonstrates the creation and reconfiguration of OPS-based VNs by updating the LUTs stored in the switching nodes. For application flows with high priority, QoS can be guaranteed with proper priority assignment in the label field avoiding the performance degradation caused by the competition. In addition, the SDN controller is able to monitor the network by collecting real-time per-port statistics through the OpenFlow protocol. The load balancing operation can be introduced to further provide the QoS support which is a valuable feature in challenging situations.

Acknowledgment

The authors would like to thank the FP7 LIGHTNESS project (n° 318606) for supporting this work.

References

[1] M. Meeker and L.Wu, "2013 internet trends," Kleiner Perkins Caufield & Byers, Technical Report (2013). [Online]. Available: http://www.kpcb.com/blog/2013-internet-trends

[2] Cisco Global Cloud Index: Forecast and Methodology 2013–2018, Cisco Systems, USA. (2013). [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.html

[3] S. Sakr, A. Liu, D.M. Batista, and M. Alomari, "A survey on large scale data management approaches in cloud environments," *IEEE Communications Surveys & Tutorials*, vol. 3, no. 13, p. 311-336, Sept. 2011.

[4] A. Hammadi, and L. Mhamdi, "A survey on architectures and energy efficiency in Data Center Networks," *Computer Communications*, vol. 40, pp. 1-21, Mar. 2014.

[5] T. Koponen, K. Amidon, P. Balland, M. Casado, A.Chanda, B. Fulton, I. Ganichev, J. Gross, N. Gude, P. Ingram, E. Jackson, A. Lambeth, R. Lenglet, S. Li, A. Padmanabhan, J. Pettit, B. Pfaff, R. Ramanathan, S. Shenker, A. Shieh, J. Stribling, P. Thakkar, D. Wendlandt, A. Yip and R. Zhang, "Network virtualization in multi-tenant datacenters," in *Proc. 11th USENIX Symposium on Networked Systems Design and Implementation*, Seattle, 2014, pp. 203-216.

[6] C.-P. Bezemer, A. Zaidman, B. Platzbeecker, T. Hurkmans and A.'t Hart, "Enabling multi-tenancy: An industrial experience report," in *Proc. IEEE International Conference on Software Maintenance (ICSM)*, Timisoara, 2010, pp. 1-8.

[7] S. Huang, B. Guo, W. Ju, X. Zhang, J. Han, C. Phillips, J. Zhang and W. Gu, "A Novel Framework and the Application Mechanism with Cooperation of Control and Management in Multi-domain WSON", *Journal of Network and Systems Management*, vol. 21, pp. 453-473, Sept. 2013.

[8] T. Dillon, C. Wu, and E. Chang, "Cloud computing: issues and challenges," in *Proc. 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, Perth, 2010, pp. 27-33.

[9] M. F. Bari, R. Boutaba, R. Esteves, L.Z. Granville, M. Podlesny, M.G. Rabbani, Q. Zhang, M.F. Zhani, "Data Center Network Virtualization: A Survey," *IEEE Communications Surveys & Tutorials*, vol.15, pp. 909-928, May 2013.

[10] A. Benner, "Optical interconnect opportunities in supercomputers and high end computing," in *Proc. Opt. Fiber Commun. Conf. (OFC)*, Los Angeles, 2012, pp. 1–60.

[11] A. Ghiasi, "Is there a need for on-chip photonic integration for large data warehouse switches," in *Proc. 9th IEEE Int. Conf. Group IV Photon.*, San Diego, 2012, pp. 27-29.

[12] L. A. Barroso and U. Hölze, "The datacenter as a computer: an introduction to the design of warehouse-scale machines," Morgan and Claypool Publishers, Los Angeles, 2009.

[13] Understanding the Benefits of Open Networking and SDN，Dell Inc., USA. (2014). [Online]. Available: http://www.dell.com/learn/us/en/555/business~solutions~whitepapers~en/documents~dell-open-networking-and-sdn-by-tech-target.pdf

[14] C. Kachris and I. Tomkos, "A Survey on Optical Interconnects for Data Centers," *IEEE Communications Surveys & Tutorials*, vol.14, pp.1021-1036, Jan. 2012.

[15] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: a topology malleable data center network," in *Proc. Ninth ACM SIGCOMM Workshop on Hot Topics in Networks*, Monterey, 2010, pp. 8:1–8:6.

[16] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. E. Ng, M. Kozuch, and M. Ryan, "c-Through: Part-time Optics in Data Centers," *ACM SIGCOMM Computer Communication Review - SIGCOMM '10*, vol. 40, pp. 327-338, October 2010.

[17] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, "Helios: a hybrid electrical/ optical switch architecture for modular data centers," *ACM SIGCOMM Computer Communication Review - SIGCOMM '10*, vol. 40, pp. 339-350, October 2010.

[18] O. Liboiron-Ladouceur, A. Shacham, B. A. Small, B. G. Lee, H. Wang, C. P. Lai, A. Biberman, and K. Bergman, "The data vortex optical packet switched interconnection network," *J. Lightwave Technol.*, vol. 26, no. 13, pp. 1777–1789, Jul 2008.

[19] X. Ye, Y. Yin, S. J. B. Yoo, P. Mejia, R. Proietti, and V. Akella, "DOS: A scalable optical switch for datacenters," in *Proc. of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, La Jolla, 2010, pp. 24:1–24:12.

[20] S. Peng, D. Simeonidou, G.Zervas, R. Nejabati, Y. Yan, Y. Shu ; S. Spadaro, J. Perello, F. Agraz, D. Careglio, H. Dorren, W. Miao, N. Calabretta, G. Bernini, N. Ciulli, J. C. Sancho, S. Iordache, Y. Becerra, M. Farreras, M. Biancani, A. Predieri, R. Proietti, Z. Cao, L. Liu, S. J. B. Yoo, "A novel SDN enabled hybrid optical packet/circuit switched data centre network: The LIGHTNESS approach," in *Proc. IEEE. European Conference on Networks and Communications (EuCNC)*, Bologna, 2014, pp. 1-5.

[21] W. Miao, J. Luo, S. D. Lucente, H. Dorren, and N. Calabretta, "Novel flat datacenter network architecture based on scalable and flow-controlled optical switch system," *Opt. Express*, vol. 22, no. 3, pp. 2465–2472, Feb. 2014.

[22] M. Channegowda, R. Nejabati, and D. Simeonidou, "Software-defined optical networks technology and infrastructure: enabling software-defined optical network operations," *IEEE Journal of Optical Communications and Networking*, vol. 5, pp. A274-A282, Oct. 2013.

[23] W. Miao, S. Peng, S. Spadaro, G. Bernini, F. Agraz, A. Ferrer, J. Perello, G. Zervas, R. Nejabati, N. Ciulli, D. Simeonidou, H.J.S. Dorren and N. Calabretta., "Demonstration of Reconfigurable Virtual Data Center Networks Enabled by OPS with QoS Guarantees," in *Proc. European Conference on Optical Communication (ECOC)*, Cannes, 2014, pp. 1-3.

[24] X. Cao, N. Yoshikane, T. Tsuritani, I. Morita, M. Suzuki, T. Miyazawa, M. Shiraiwa, and N. Wada, "Dynamic Openflow-Controlled Optical Packet Switching Network," *J. Lightwave Technol.*, vol. 33, no. 8, pp. 1500-1507, April 2015.

[25] W. Miao, S. D. Lucente, J. Luo, H. Dorren, and N. Calabretta, "Low latency and efficient optical flow control for intra data center networks," *Opt. Express*, vol. 22, no. 1, pp. 427–434, Jan. 2014.

[26] OpenDayLight project, http://www.opendaylight.org/

[27] OpenFlow, https://www.opennetworking.org/sdn-resources/openflow

[28] S. Das, "Extensions to the OpenFlow protocol in support of Circuit Switching". [Online]. Available: http://archive.openflow.org/wk/images/8/81/OpenFlow_Circuit_Switch_Specification_v0.3.pdf