An integrated SDN-based architecture for Passive Optical Networks

Hamzeh Khalili, David Rincón, Sebastià Sallent, José Ramón Piney

Dept. of Network Engineering, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{khalili.hamzeh, drincon, sallent, jpiney}@entel.upc.edu

Abstract: Passive Optical Network (PON) are often managed by non-flexible, proprietary network management systems. Software Defined Networking (SDN) opens the way for a more efficient operation and management of networks. We describe a new SDN-based architecture for Ethernet Passive Optical Networks (EPON), in which some functions of the Optical Line Terminal (OLT) are virtualized and located in an external controller, while keeping the rest of the PON functionality around an Open Flow switch. This opens the way for an improved management of the resource usage, bandwidth allocation, Quality-of-Service (QoS) monitoring and enforcement, or power consumption management, among other possibilities. In order to maintain the time-sensitive nature of the EPON operations, synchronous ports are added to the switch. OpenFlow messages are extended in order to cope with the PON-related parameters. Results based on simulations demonstrate that our proposal performs similarly or better than legacy architectures, in terms of delay and throughput.

Keywords: Software Defined Networking (SDN), OpenFlow Protocol, Passive Optical Network (PON), Ethernet Passive Optical Network (EPON), Service Interoperability for EPON (SIEPON).

1. Introduction

1.1. Ethernet Passive Optical Networks

Passive Optical Networks technologies (among them, EPON [1] and GPON [2]) are currently used in access networks. As shown in Figure 1, PON is composed of a central office (Optical Line Terminal, OLT), passive optical splitters, and one terminal (Optical Network Unit, ONU) at each one of the customer premises, in a tree topology. First-generation PONs use Time Division Multiplexing (TDM) for sharing the medium in a point-to-multipoint scenario. In the downstream, the OLT broadcasts the data frames towards the ONUs, while in the upstream an arbitration mechanism is needed in order to avoid collisions when ONUs send frames to the OLT. The EPON standard, which will be the focus of this work, uses the Ethernet format for the data frames.

A key piece in EPON is the concept of Dynamic Bandwidth Allocation (DBA) algorithms, which allow the OLT to orchestrate the access to the shared medium and to dynamically adapt to the changing requests from the ONUs. In the EPON architecture, an entity called Multi Point Control Protocol (MPCP) manages the upstream channel and harmonizes the transmission of data from ONUs to OLTs. In addition, it manages operations such as terminal discovery, registration, and bandwidth allocation. MPCP uses two standard messages for DBA operations, namely GATE and REPORT. REPORTs are used by ONUs to request transmission opportunities, and GATEs are sent by the OLT to grant a transmission slot (time and length) to a specific ONU.

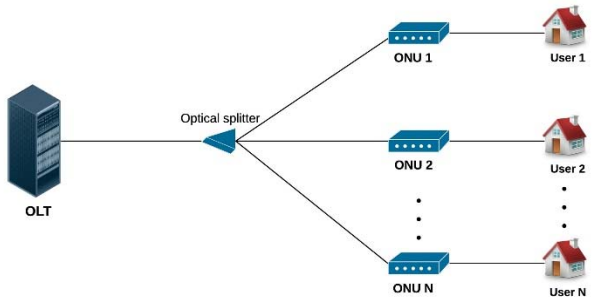

Figure 1. PON topology

1.2. EPON Service Interoperability

In order to ensure service interoperability in a multivendor scenario, IEEE developed the 1904.1 Service Interoperability for Ethernet Passive Optical Networks standard (SIEPON) [3][4][5]. One of the key concepts in SIEPON is the definition of an unified data path architecture to ensure that data services can be managed, provisioned and monitored across the EPON. SIEPON covers the physical (PHY), Medium Access Control (MAC), and upper layers to manage tasks related to the data path, such as multicast delivery, tunnels, VLANs, and Quality-of-Service (QoS) enforcement.

EPON Service Pathways (ESPs) are defined as the path that a data frame follows through the Media Access Control (MAC) client functional blocks, specifying both the connectivity and QoS of the frame. Connectivity is defined by classifying the frame according to header fields (VLAN tag, MAC address, IP address, to name a few possibilities), while QoS parameters can be enforced by queues with priorities and scheduling. Figure 2 shows the SIEPON architecture in both the OLT and the ONUs, and specifically the ESP-related blocks: the Input block [I] is the ingress port that receives frames from User-Network Interface (UNI) or Network-Network Interface (NNI); the Classifier block [C] classifies incoming frames; the Modifier block [M] is able to modify frame fields; the Policer/Shaper block [PS] enforces conformance to the service contract; the Cross-connect block [X] is used to direct a frame to the proper queue; the Queue block [Q] holds frames until they are selected by the scheduler for transmission; the Scheduler block [S] transmits the frames from the queue block to the output block using a predefined scheduler algorithm; and finally Output block [O] is the egress port that receives frames from a scheduler and forwards them to the UNI or NNI. Each one of the ESP blocks can involve several independent instances operating on various flows of traffic.

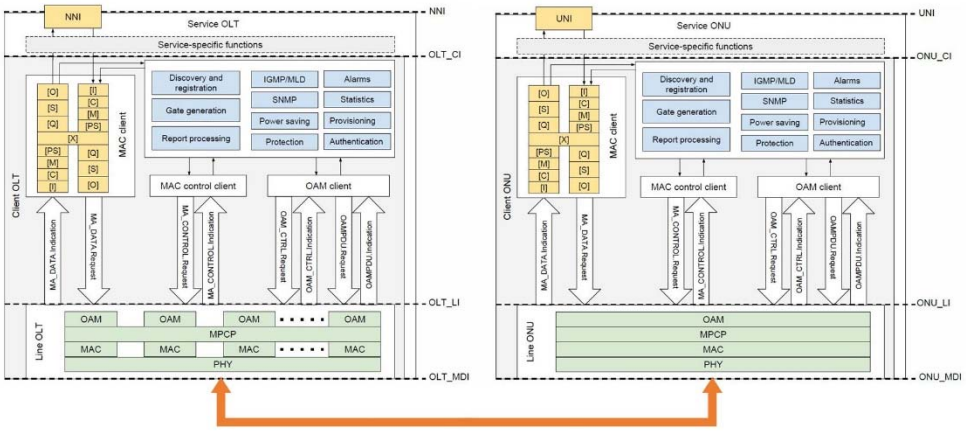

Figure 2. SIEPON architecture, OLT (left) and ONU (right)

1.3. Software Defined Networking

Software Defined Networking (SDN) [6] is a new paradigm for network management that introduces network programmability and separation of the data and control planes, migrating the complex, "intelligent" functions from the network devices to a centralized controller. These characteristics allow the SDN controller to obtain a centralized and accurate view of the state of the network, thus opening the way for the optimization in several ways (bandwidth assignment, QoS guarantees, minimization of power consumption, or resilience, to name a few possibilities).

The SDN architecture defines two interfaces in the controller. The northbound interface defines an API that allows network management systems (NMS) or ad-hoc applications to communicate with the controller. The southbound interface communicates the control plane that runs in the controller with the data plane that resides in the network equipment (switch). The original southbound API, and probably the most popular, is OpenFlow (OF). The OpenFlow protocol [7] defines a standardized instruction set that allows the controller to manage any OpenFlow-enabled device and specify the path to be followed by traffic flows through the network of switches, by means of the definition of matching rules (based on packet header fields) and operations to be performed (e.g. forwarding to a specified port, modifying the headers, or dropping the packet).

1.4. SDN-based PON networks

The management of access and backbone optical networks is becoming more and more important in the research literature [8], and how to apply SDN is being now intensively investigated. In the specific case of PON networks, given its centralized nature (the OLT manages almost all the operations), applying the centralized control envisioned by SDN makes even more sense. However, the aspects related to service interoperability, or a detailed SDN architecture (including extensions of OpenFlow messages and actions), are relatively unexplored. The authors of [9] provide a useful and up-to-date survey on the contributions on SDN- based optical networks. Here, we discuss the most relevant works in the field of SDN and optical access.

In [10] and [11], authors envision a scenario where each optical switching node is virtualized, in order to obtain a unified control plane. Each physical interface is mapped to a virtual interface. In order to ensure a smooth transition from legacy equipment, an extra layer would translate the messages between the controller and the switches [12], at the cost of adding latency by using message proxies. The authors of [13] developed an architecture for PON networks based on SDN including an OpenFlow extension for traffic mapping and forwarding capabilities, with no effect on data link layer latency. In [14] an architecture control plane for converged metro-access networks under SDN is described. [15] presents a novel software-defined optical access network (SDOAN) architecture. The purpose of developing a Service-Aware Flow Scheduling (SA-FS) strategy is to assign network bandwidth resources in an efficient and flexible way. In order to operate PONs coordinated with a core SDN-based network, [16] describes an architecture for a Software-Defined Edge Network (SDEN) able to control end-to-end the traffic flows. In [17], authors propose a new SDN- and NFV- (Network Function Virtualization) based architecture for EPON networks; in this scheme, OLTs and ONUs are partially virtualized and moved to a central controller. [18] describes, at a high level, an EPON architecture based on SDN that replaces the hardware-based DBA with a software-based one, residing in the controller. The authors of [19] describe a GPON architecture where an OpenFlow agent is located in the OLT to communicate with the SDN controller, claiming that the approach can connect several sites in different locations in a cost effective manner.

Central Office Re-architected as a Datacenter (CORD) is a novel architecture developed in[20] aimed at substituting the telephone exchange hardware with software-based equipment, converting central offices into datacenters, with the purpose of speeding the deployment and increasing the efficiency of services. CORD decouples the control and data planes, using the Open Network Operating System (ONOS) [21] SDN controller, and virtual machines running on top of OpenStack. CORD is currently supported by service providers such as AT&T and NTT Communications.

To the best of our knowledge, [22] and [23] are the only works that tackle the topic of how to adapt the SIEPON standard to the SDN architecture. [22] describes methods for enabling nodes to exchange information between a management protocol and other protocols, including control plane and data plane interfaces to support SDN. [23] expands the work presented in [17], converting the OLT into an OpenFlow switch, and describes the extensions to make it compatible with the SIEPON architecture, and thus reach the goal of having a virtualized, simplified and centralized management system migrated to the SDN controller.

The remainder of this chapter is organized as follows: In Section 2, we describe the SDN-based EPON architecture and the OLT functions. Section 4 expands the description by defining the new elements and modifications, together with details of its operation and implementation. Section 4 presents the validation of the proposal, together with results obtained with a simulator and a reference implementation. Section 5 concludes the chapter.

2. SDN-SIEPON architecture

This work builds on [17] and expands [23] with more details and results. We present a novel SDN-SIEPON architecture based able to decouple, virtualize, and simplify the management and operation of the OLT and OAM functions, and opens the way for multi-tenant and multi-provider optical access networks, where various service providers use the same basic infrastructure.

Our architecture is based on the principle that the OLT is built around an OpenFlow switch that takes care of the forwarding plane and carries out the functions related to the EPON service path, while some of the control plane-related functions are migrated to an OpenFlow controller. The OpenFlow switch performs the following functions: (1) classifies incoming packets following the matching rules; (2)

modifies packet header fields if required; (3) schedules and ensures QoS for each flow; and (4) forwards packets toward their destination ports. In order to do so, the OpenFlow switch emulates some of the EPON functionalities defined in the OLT, such as Control Multiplexer (CM), Control Parser (CP), and Multipoint Transmission Control (MPTC). Many of these functionalities work in real time, and therefore we need to define synchronous ports and make it compatible with the SDN architecture by defining a set of registers. The operations that must be executed in real-time are kept in the switch, while those that work at longer time scales can be migrated to the SDN controller.

As shown in **¡Error! No se encuentra el origen de la referencia.**, the SDN-SIEPON architecture includes three main elements: an SDN-based OLT; an SDN controller; and the EPON network, including the passive splitters, the fibres and the ONUs. We focus here on the redesign of the OLT, and although the partial virtualization of the ONUs might be of interest (for example, in the context of energy saving or QoS management), it is out of the scope of this work and left for future research.

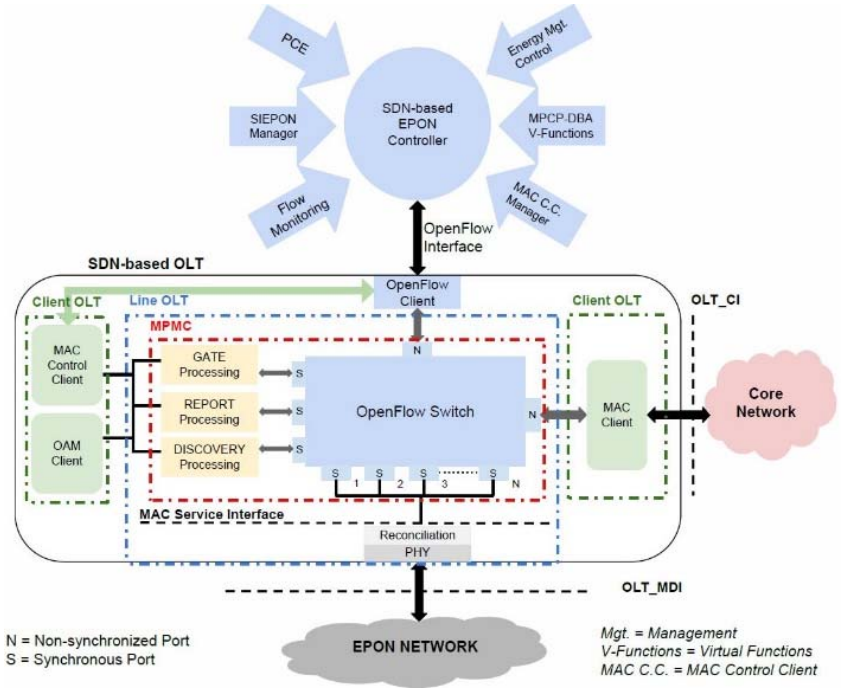

Figure 3. Elements of the SDN-SIEPON architecture

2.1. SDN-OLT

The SDN-based OLT is built around an OpenFlow switch, and its tasks include the forwarding functions of the MPMC sub-layer and the operations of the Control Multiplexer, Control Parser, and Multipoint Transmission Control elements of the EPON architecture.

A non-synchronous port is in charge of the connection of the SDN-OLT to the SDN controller, while a set of synchronous logical ports link the ONUs and the operations of the SDN-OLT (among them the processing of the GATE and REPORT messages, the discovery process, and the DBA management).

There is a synchronous port per ONU (with a maximum of 64 ONUs per OLT), attached to a specific MPMC instance running in the SDN-OLT, plus one extra logical port for broadcasting messages to all ONUs. The 64 port-instance pairs are activated on demand, depending on the number of active ONUs. Whenever an ONU is activated, the SDN controller creates a MPMC instance and links its three logical ports to the DISCOVERY, GATE, and REPORT processing modules. In parallel, the controller creates a unicast MAC service logical port and connects it with the recently started up instance. While legacy OLTs require several complex entities with multiple coordinating entities, our design simplifies the architecture by setting the OpenFlow switch as the single coordinator of all the entities, thus arbitrating the access to the downstream and upstream channels, as illustrated in Figure 4.

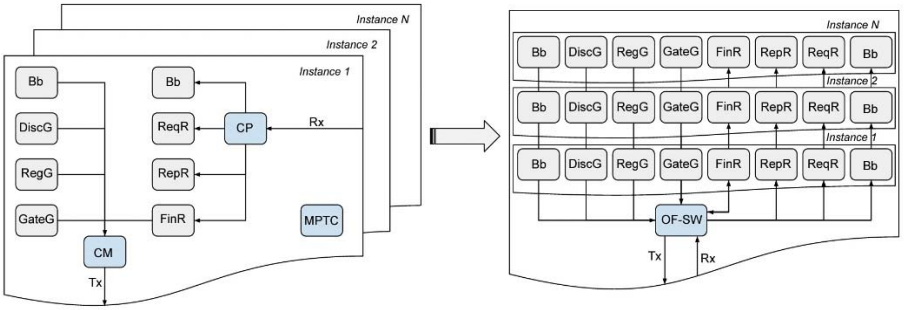

Figure 4. Left: architecture of a legacy OLT with several complex MPMC instances. Right: the simplified SDN-OLT architecture. Acronyms of the processing entities: OF-SW (OpenFlow switch); Bb (backbone), DiscG (discovery gate generation); RegG (register generation); GateG (gate generation); FinR (Final Registration); RepR (report reception); and ReqR (request reception).

2.2. SDN controller

The SDN controller is a piece of software that manages one or several SDN-OLTs, and runs the complex, non-real time functions that compose the control plane of the EPON network. It is usually run on virtual machines in a data centre, and has a centralized view of the network. One key aspect of our design is the decoupling of the virtualizable functions that work in longer timescales and can be migrated to the controller from the functions that work at shorter timescales and have to be kept at the switch. To name a few cases, the controller manages the parameters of the MAC control client; the policies of the MAC client; or the translation of the linking the high-level SIEPON QoS class definition to the low-level MAC QoS parameters.

An important SIEPON module that is also migrated to the controller is the ESP manager of the MAC client, responsible of aspects such as provisioning, QoS policies, flow management, bandwidth assignment policy, time-related decisions, addition of unregistered ONUs to the network, and data forwarding services through the SDN-OLT. It is also in charge of handling and configuring the initial parameters of the MAC control client in the OpenFlow switch, and the processing of GATE, REPORT, and DISCOVERY messages as well. It also provides a database for the OpenFlow switch to keep information regarding logical link identifiers (LLIDs), status of logical port, and ONU MAC address so as to configure flow tables for the switch.

The MPCP-DBA virtual module handles the overall network policy function and allows the controller to communicate with SDN-based OLT to alter parameters and policies of DBA, ONU priority in DBA, or shift between several available DBA algorithms.

3. Extensions and modifications of OpenFlow

This section describes the extensions and additions to the OpenFlow architecture that are needed for our proposal. OpenFlow messages and actions will have to be extended to support EPON operations, and synchronous ports will be needed in the switch in order to keep the synchronous nature of EPON. Another aspect to tackle is how to implement the ESP functional block of SIEPON. Finally, a new and simpler MPMC sub-layer will be described.

3.1.1. Extension of OpenFlow messages

The original OpenFlow messages lack specific headers and parameters needed to execute some of the SIEPON-related operations. Basically, we need:

1) A way to determine whether the OF switch is a SDN-OLT type or not. This is solved with an extension of the *OFPT_FEATURES_REPLY* message sent in response to the controller query.
2) Maintain correspondence of the Logical Link identifier (LLID) of the port and the MAC address of the ONU. This is done by extending the *OFPT_PORT_STATUS* message.
3) Set up the initial parameters of the MAC control client and MAC client (for example, setting the LLIDs) during the initial phase of the dialog between the SDN-OLT and the SDN controller. This can be done by extending *OFPT_SET_CONFIG* message.
4) Transport the set of synchronous rules, matching fields and actions. This information is carried in an extension of the *OFPT_FLOW_MOD* message.

OpenFlow match fields must also be extended in order to support the new SIEPON-related functionalities. A *PDU_type* field is created to maintain the type of received packet (a MAC control frame or a data frame). An *opcode* type distinguishes the control frames as a DISCOVERY GATE, REGISTER_REQ, REGISTER, REGISTER_ACK, GATE or REPORT message. *Flag* is used to classify the type of GATE packet (DISCOVERY GATE or normal GATE). *LLID* contains the port's LLID number, which is given by the controller and allocated by the MAC control client. Finally, a *Grants_number* stores how many grants have produced as result of the processing of the GATE message. It can range from 0 to 4, where 0 signals a periodic GATE packet and 1 to 4 are related to the bandwidth allocation by a normal GATE.

3.1.2. Synchronous ports

The synchronous nature of PON networks requires the SDN-OLT to be able to handle real-time operations in synchronous ports, through which packets are sent or received in specific times specified by the synchronous flow entries. We developed a MPMC sub-layer extension that retains synchronicity and allows us to replace Control Parser, Control Multiplexer, and Multipoint Transmission Control components of a legacy OLT with the OpenFlow switch.

The synchronous operations related with a newly added ONU include:

- Setting an instance_ID in the OpenFlow switch for the respective MPMC instance.
- Allocating three synchronized logical ports to each gate, report, and discovery processing instance.

- Allocating an input/output synchronized logical port with an associated LLID so the port is linked with the MAC address of the newly connected ONU.
- Timestamping of packets delivered to/received from the ONUs. The timestamps are obtained from the local clock of the OpenFlow switch.
- Computing and monitoring the round trip time (RTT) of the packets received from the ONUs.
- Assigning bandwidth and transmission time slots in the upstream direction (packet forwarding strategy), in cooperation with the SIEPON manager (based on its QoS policies).
- Creating and destroying flow entries in the OpenFlow switch.

Several registers are used to emulate the synchronous operation:

- P (transmitPending), I (transmitInProgress), and E (transmitEnable) are employed to synchronize the MPMC instances with the DBA (thus allowing the packets from the instances to share the channel in an arbitrated way). The OpenFlow switch controls the transmission of packets towards the Reconciliation Sub-layer (RS) by activating the transmitEnable signal.
- Two additional registers are used for round-trip-time (RTT) calculation and packet timestamping. RTT calculation is used for synchronization between OLT and ONU.

Figure 5 illustrates the extension of the MPMC sub-layer of the SDN-OLT.

3.1.3. ESP functional block

The aforementioned extension of the Flow-Mod message allows us to introduce the functionality of the SIEPON ESP functional blocks in the SDN-OLT. Figure 6 describes an example of an OpenFlow flow entry extended with the new matching fields and actions. The actions related with the ESP functional blocks are:

1. Set-Field: it is consistent with the Modifier block [M] and implements all the set-field actions in the packet. It modifies the value of the packet header fields (e.g. Ethernet/IPV4/IPV6 source and destination addresses, VLAN ID and VLAN priority, for example).
2. Apply meter: it is consistent with the Policer/Shaper block [PS] and allows packets' rate measurement and control (setting rate limit), and implement DiffServ-like policy. In addition, meters can assist per-port queues in scheduling packet on an output port with respect to their priority, thus ensuring QoS.
3. Set-Queue: it is consistent with the Queues block [Q], and sets the queue_id for a packet when it is delivered to the port. It is used for packet scheduling and forwarding.
4. The Scheduler block [S] controls the actions related with the transmitPending, transmitInProgress and transmitEnable actions by altering their associated registers P, I, E.
5. Output port: it is consistent with the Output block [O]. Packets are allocated to a physical port (i.e. controller) or a logical port (either unicast or broadcast, depending on the ONUs targeted).
6. Operations performed by the Classifier [C] and Cross-connect [X] blocks are executed via rule matching and actions in the flow table. Incoming packets are compared and classified based on their matching fields with the existing rules, and packets are later sent to the destination based on the associated action.

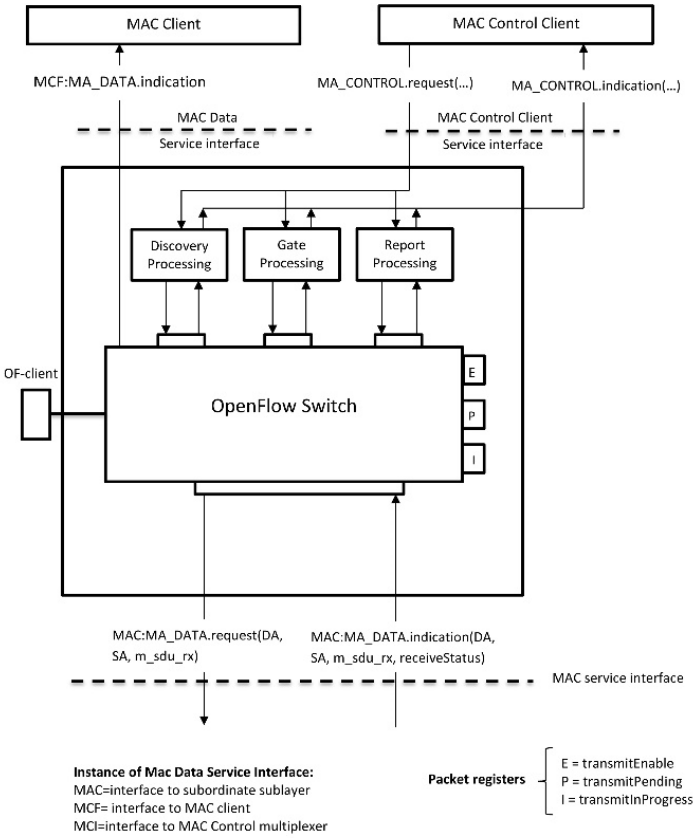

Figure 5. MultiPoint MAC Control functional blocks, from [23].

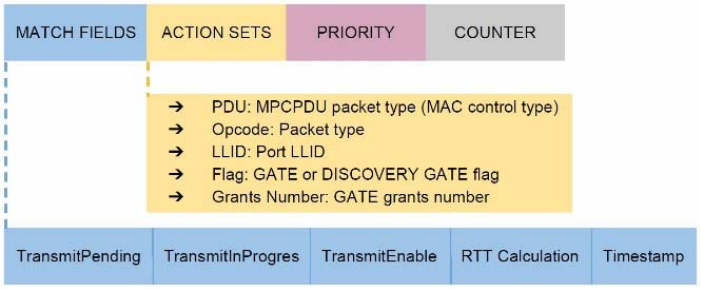

Figure 6. Example of the extended match fields and actions.

3.1.4. MPMC sub-layer

In the legacy OLT the MPMC sub-layer is quite complex: a MPTC module synchronizes multiple instances of MPMC (one for each ONU, with a Control Multiplexer and a Control Parser entity). Our proposal simplifies this by introducing three single entities of MPTC, CM, and CP, implemented as rules and flow entries in the OpenFlow switch. Therefore, the switch's task is to coordinate with different MAC instances of the OLT and coordinate their requests for sending or receiving packets.

Figure 7 illustrates the main blocks that compose an OpenFlow switch: OF agent; control path; and data path. The OpenFlow agent interacts with the SDN controller in order to manage both the control and data paths. The control path is responsible of emulating the OLT functionality through the set of rules and actions in the flow entries. The data path task is to match the received packets with respect to the matching rules, and execute the attributed actions (e.g. frame forwarding and scheduling for transmission).

The MPMC sub-layer carries out the following functions: after receiving a frame from the underlying MAC, forwards it to the OpenFlow switch flow table in which the frame is analysed based on its opcode. The REPORT and data frames coming from ONUs are analysed by CP functions. In Fig. 7 each data path flow entry related to the control path element is shown in the same colour. The MPMC also coordinates the processing of DISCOVERY GATE, GATE, and data frames created by the MAC instances and forwards them to the RS layer (shown in green in Fig. 7), and manages the transmission of data frames using the three registries (i.e. P, I, and E) defined in the OpenFlow switch control path.

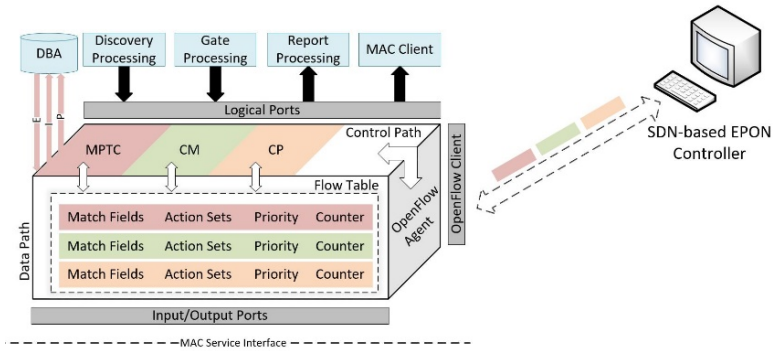

Figure 7. MPMC sub-layer design.

3.2. An example of operations

We now provide an example to illustrate the procedure for generating a new flow entry in the SDN-OLT for processing a DISCOVERY GATE, GATE and REPORT messages. Let's recall that a GATE message tells a specific ONU when to send data (start time) and how much data to send, in response to a previous REPORT message from the ONU requesting a transmission opportunity. Regarding the discovery process, DISCOVERY GATE messages are broadcasted periodically to all ONUs, and are replied by the newly connected ONU with a REGISTER REQUEST (randomly delayed in order to minimize collision with other ONUs). The OLT confirms by sending a REGISTER message immediately followed by a GATE that grants and schedules an ONU response in the form of a REGISTER-ACK message, finishing the discovery and register process. **¡Error! No se encuentra el**

origen de la referencia. illustrates the state diagram for the actions related to processing the aforementioned messages.

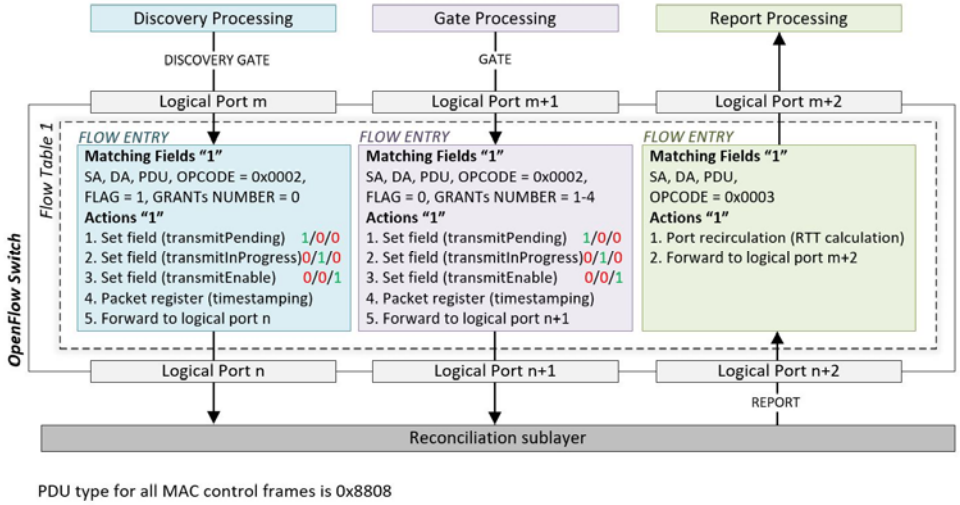

Figure 8. Operations related to the processing of DISCOVERY GATE, GATE and REPORT messages.

When the SDN-OLT establishes the communication with the SDN controller, the latter sends a series of FlowMod messages with pre-defined rules to be inserted in the flow table of the OpenFlow switch. These rules define how to process the DISCOVERY GATE, REGISTER_REQ, REGISTER, REGISTER_ACK, GATE, and REPORT messages. MAC control client sends a request to a discovery processing entity of an instance to creates a DISCOVERY GATE message with the required fields (discovery information and policy to use for the ONU) and sends it through the appropriate synchronous logical OF switch port. After receiving a packet, a flow table lookup is performed by comparing the packet header fields with the matching fields of the installed rules by observing a priority order. When the packet is matched, the attributed counter for the chosen flow entry is increased and the actions linked with the flow entry are performed.

When, later on, the controller receives a request (through Packet-In) from the SDN-OLT, for processing a GATE message, it gathers and analyses the received packet data (source and destination MAC addresses, PDU type, opcode, flag, grant number, LLID and VLAN IDs) to set a rule in the OF switch for determining the grants' number and establish the QoS policy, and prioritize packet delivery toward an output port. The rule is transmitted through a FlowMod and a Packet-Out messages to the OpenFlow switch, and the received GATE message is processed accordingly.

For both the DISCOVERY GATE (right side of **¡Error! No se encuentra el origen de la referencia.**) and GATE (centre of **¡Error! No se encuentra el origen de la referencia.**) messages, when the packet is prepared for broadcasting, the packet is scheduled for transmission by setting register P (transmitPending). The packet, placed in the queue, might wait for higher priority packets to be processed by the DBA. The transmitInProgress action is performed by setting register I so as to alter transmission state of the packet to transmitInprogress (by setting I to 1 and transmitPending to 0). Then, transmitEnable action is performed when no more packets are available to be transmitted by setting register E, and the transmission state of the packet is altered to enable (E is set to 1 and

transmitInProgress to 0). The next action is to send the packet toward a specified logical output port. A timestamp is then added to the frame and the packet is sent to the RS layer. In the case of REPORT message (at the left side of **¡Error! No se encuentra el origen de la referencia.**), as soon as the REPORT message is received from an ONU and matched against a rule in the flow table, the action of calculating the RTT is executed on the packet (via port re-circulation and interacting with the packet registers) and the value is communicated to the DBA. Once the RTT is calculated, the packet is forwarded to the REPORT entity for further processing.

4. Evaluation

This section describes the simulation model and the performance evaluation of our proposal. We built models of both the legacy EPON architecture of the SDN-based EPON architecture using the OPNET Modeler package, and compared them in terms of delay, and throughput. The results expand our initial validation described in [23] with an evaluation of the influence of the traffic pattern, the number of active ONUs, and end users in the performance of the system.

4.1. Description of the scenario and delay analysis

A scenario with a SDN controller, a SDN-OLT, and a tree topology with 16 ONUs was built. The distance between the SDN controller and the OLT is initially set to 1 km, and the ONUs are at distances ranging from 16 to 18 km from the OLT. The link rate in both the downstream and upstream directions is 1 Gbit/s, the guard time is 1 μs, and the maximum cycle time is set to 1 ms. IPACT [24] is used as DBA algorithm, and the message processing delay in the SDN-OLT is set to 0.0164 ms, [25].

Two traffic generators are used: constant bitrate and self-similar traffic [26, 27]. For CBR traffic, packets have a constant length of 791 bytes, while in the second case packet size is uniformly distributed between 64 and 1518 bytes (with a mean of 791 bytes, to compare with the CBR case), and the Hurst parameter (a measure of the long-range dependence of self-similar traffic) was set to 0.7, 0.8, and 0.9. In order to understand the delay analysis that follows, **¡Error! No se encuentra el origen de la referencia.** shows the scenario and the delay measurement points, where the red path is used for calculating the round trip time (RTT) delay, and the DE path is used for evaluating the queueing delay and throughput of the system.

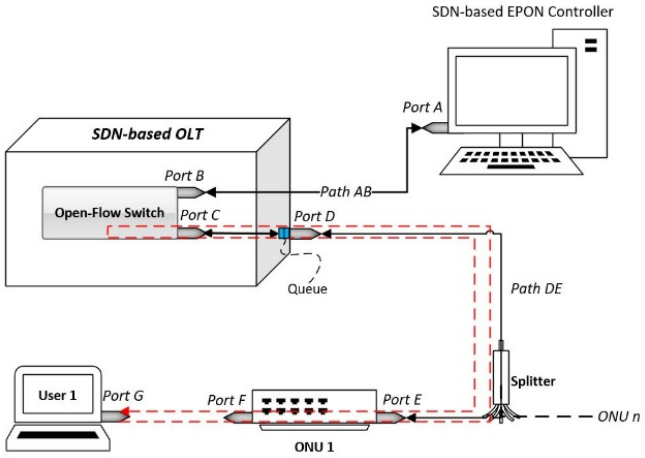

Figure 9. Measuring points in the SDN-based EPON scenario.

Table 1 shows the time it takes for the controller to process each type of packet, and the time for transmitting the packet through the logical output port in the SDN-OLT, for the cases of the six initial FlowMod messages (DISCOVERY GATE, REGISTER_REQ, REGISTER, REGISTER_ACK, GATE, and REPORT,) where the OpenFlow installs the flow rules at the switch during the connection establishment. The average packet delay is measured for two different cases of distance between the controller and the SDN-OLT, 1 Km and 10 Km. Even for the longer distance, in which a single controller located in a data centre could manage simultaneously several OLTs in a diameter of 20 Km (i.e, a medium-sized city), response times are almost negligible.

	Tx delay in the controller (port A)	Rx delay in the switch (port B), d = 1 km	Rx delay in the switch (port B), d = 10 km	Tx delay through path AB, d = 1 km	Tx delay through path AB, d = 10 km
DISCOVERY GATE	0.52 µs	5.76 µs	51.66 µs	5.243 µs	51.14 µs
REGISTER REQ	1.04 µs	6.28 µs	52.18 µs		
REGISTER	1.56 µs	6.80 µs	52.70 µs		
REGISTER_ACK	2.08 µs	7.32 µs	83.22 µs		
GATE	2.60 µs	7.84 µs	83.70 µs		
REPORT	3.12 µs	8.36 µs	54.26 µs		

Table 1. Average values of packet processing delays.

A metric of interest is the influence of the presence or absence of the flow processing rules in the performance of the SDN-OLT. In the first case we assume that the SDN controller has previously installed in the switch the rules for processing the control frames (DISCOVERY GATE, REGISTER_REQ, REGISTER, REGISTER_ACK GATE, and REPORT). The delay is evaluated from the moment a packet (such as control frame or data frame) enters the OpenFlow switch via a synchronous port, including the look-up for a rule in the flow table to match and perform a group of actions (such as read and write a register, and forward the packet to the output port). The average processing time in presence of the rules in the flow table is $T_{presence\ of\ rule} = 0.488\mu s$.

For the case when the rules are not yet installed in the switch, whenever a new packet arrives, the OpenFlow switch sends a Packet-In message to the controller, who in turn creates a FlowMod and a Packet-Out message to insert a rule in the flow table, followed by the look-up time and action execution. In this case, the average processing time can be described as the first look-up time ($T_{1st\ look-up} = 0.297\mu s$), plus forwarding time of the packet to the logical port B ($T_{forwarding-to-port-B} = 0.096\mu s$). In addition, the packet forwarding time from logical port B to a physical port A is $T_{AB/BA} = 5.243\mu s$, and the controller processing time for analysing the incoming the Packet-In message, installing the rule, and issue a FlowMod and Packet-Out messages is $T_{controller\ processing} = 1.986\mu s$. The second look-up and actions execution times are, as in the first case, $T_{2nd\ look-up+actions} = 0.488\mu s$. Therefore, the average processing time in absence of the rules is $T_{absence\ of\ rule} = 13.353\mu s$. Since the look-up time process will vary depending on the length of the flow table and the number of flow entries, we carried some experiments. For both cases (in presence and absence of the rules) the average look-up time when 10 flow entries are present in the table is $0.294\mu s$, increasing to $0.398\mu s$, when 100 flow entries are present. Therefore, the influence of the number of flow rule entries does not play an important role in the delay performance of the system.

4.2. QoS performance evaluation

We now focus on the evaluation of the QoS in the path between the OpenFlow switch and the end users, highlighted in red in **¡Error! No se encuentra el origen de la referencia.**. This path includes the look-up and rule matching times, registers read and write operations, and waiting time in the port queue (point D in the MAC service interface, blue queue in **¡Error! No se encuentra el origen de la referencia.**). The results are obtained by averaging 50 values for each parameter.

Figure 10 shows the round-trip-time (RTT) delay against the offered load for both the SDN-based and legacy architectures. The RTT delay is evaluated for the data frames, which are generated and forwarded to the logical port C of the OpenFlow switch by the end user through the physical port G, and returned back. In a real situation, packets would actually ingress the core network, but since we are interested in the PON segment, we do not include this term. The results are presented using box plots, showing the average delay value (center of the box), the first and third quartiles (bottom and top of the box), lowermost and uppermost values (whiskers below and above the box) and the low min and high max values (red dots scattered). Please note that, although they are related to the same x-axis value, red dots not coincided with the blue dots at the same x-axis position in order to enhance visualization and avoid overlapping. Both SDN-based and legacy EPON architectures are evaluated in the same conditions (global offered load) and each blue or red dots in the figures relates to the average value obtained for all the ONUs. The offered load is expressed as 10% fractions of the maximum capacity of the system.

The main conclusion from the left side of Figure 10 is that the delay performance is much better in the SDN-based architecture, particularly when the network load is high. The almost constant RTT in the SDN-OLT scenario is explained by to the presence of pre-defined rules in the flow table, thus ensuring

almost constant processing time. In the SDN-based architecture the DBA procedures are highly simplified, while in the legacy EPON architecture, several Control Multiplexer and Control Parser entities (one per instance) are involved. This also explains the higher variability (red dots dispersion) in the legacy case, compared with the more deterministic behaviour of the new architecture. The traffic pattern does not have a significant affect in the results. CBR and self-similar traffic offer the same average results, with less variance in the case of constant traffic. Long-range dependence characteristics of the traffic pattern do not seem to affect significantly the performance, with only a slight increase in the dispersion of the measurements when H is increased (probably related to the presence of more time-correlated traffic bursts when the Hurst parameter is increased).

The right column of Figure 10 shows the queuing delay in the downstream direction against the offered load for both architectures and for different traffic sources (self-similar with different Hurst parameters and CBR). The queuing delay is evaluated for both control and data frames at the OLT physical port (port D of the MAC service interface), and correspond to measurements of the packet waiting times in the downstream direction of the queue. The delay is measured from the moment when a packet is forwarded to the transmitter channel queue and until the time the last bit of the packet is transmitted. Our goal here is to evaluate the possible impairments in the downstream transmission in the SDN-based architecture. As discussed earlier, the OpenFlow switch affects the downstream transmission by controlling the synchronous logical ports that communicate with the DBA entity. The results show a behaviour similar to that of the RTT. In the legacy case, the queueing delay is very dependent on the load, while it almost does not vary in the case of the SDN-based architecture – and in this case the delay is much smaller. Again, this is caused by the simplified architecture of our proposal

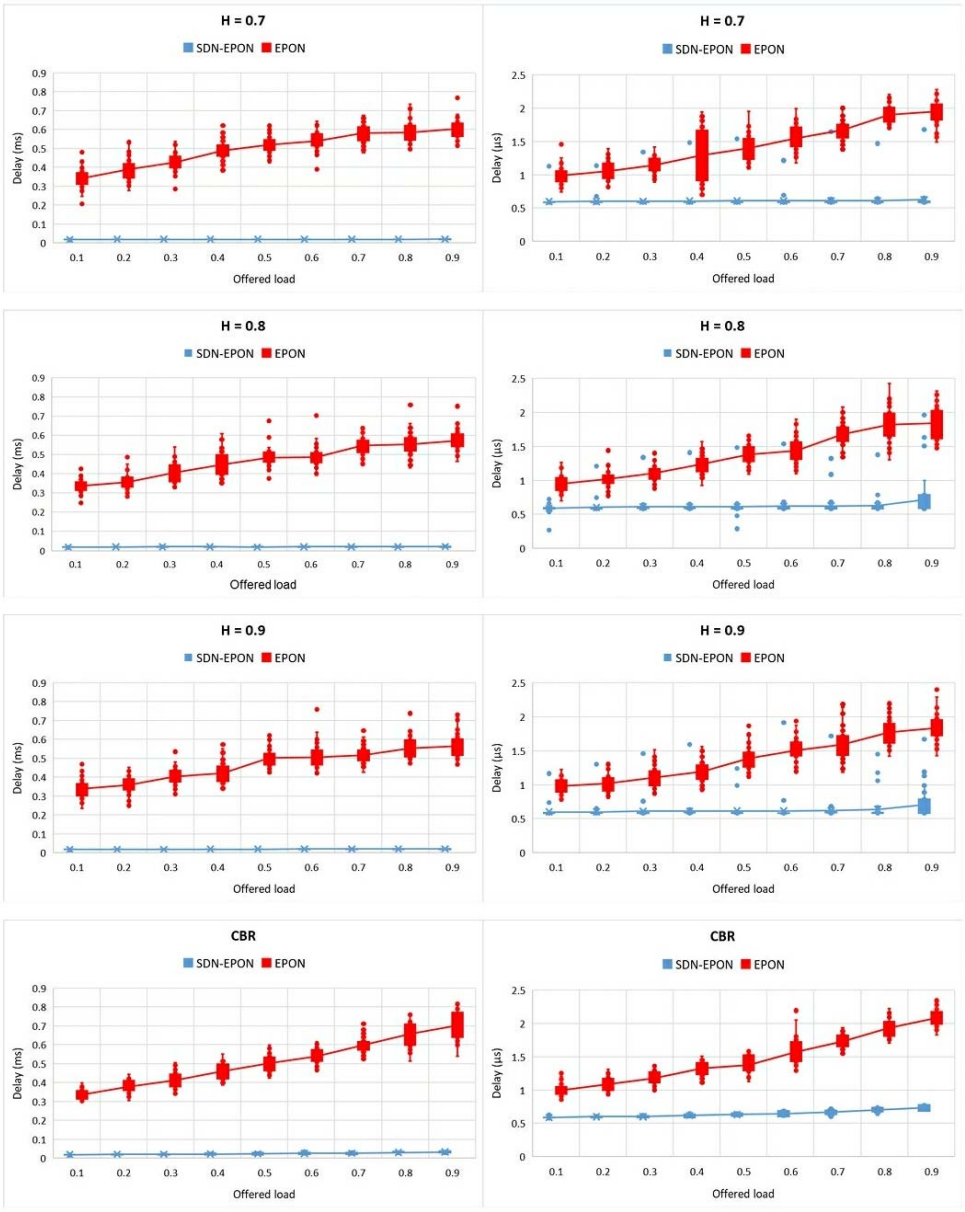

Figure 10. Comparison of RTT delay (left column) and downstream queueing delay (right column) in SDN-based EPON and legacy scenarios under different offered load, for CBR and self-similar traffic patterns and different Hurst parameter.

In the next set of experiments we vary the number of active ONUs (sixteen and eight), for different offered loads, and Hurst parameter 0.9. The case of sixteen active ONUs is the base case already analysed previously, while in the second scenario only eight ONUs send and receive the total traffic of the network (we double the rate of each ONU, in order to maintain the global load equivalent) and the other eight ONUs do not generate any user-related traffic and only communicate with the OLT through the control frame messages. As Figure 11 illustrates, in both architectures, the RTT delay increases with the input traffic, but as in the previous experiments, the delay for the legacy EPON case is much higher. The number of active ONUs has a limited influence, shown as a small increase in the delays experienced in both architectures. The increase is caused by the fact that we are averaging the delays experienced by the packets generated by ONUs that generate a rate that has been doubled, and therefore their packets experience increased waiting times.

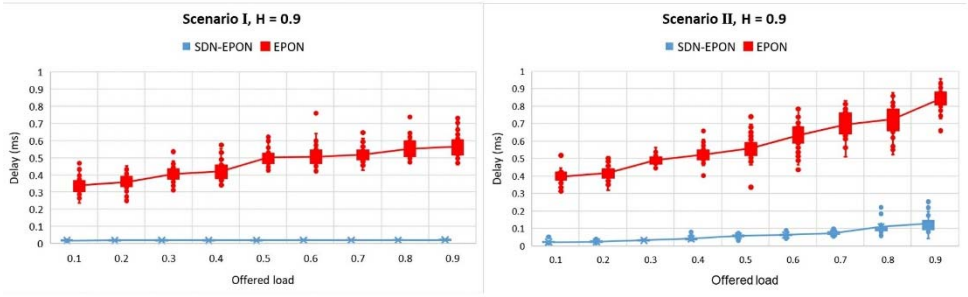

Figure 11 RTT delay in SDN-EPON and EPON architectures under different number of end users (sixteen active ONUs on the left, eight active ONUs on the right), for different offered traffic load.

The same phenomenon is seen in the queueing delay analysis, whose results are shown in Figure 12. As the number of packets per second increases at each ONU, the queueing delay increases too, but much faster for the legacy EPON scenario, and both scenarios suffer a slightly higher delay when the load is concentrated in a few the ONUs.

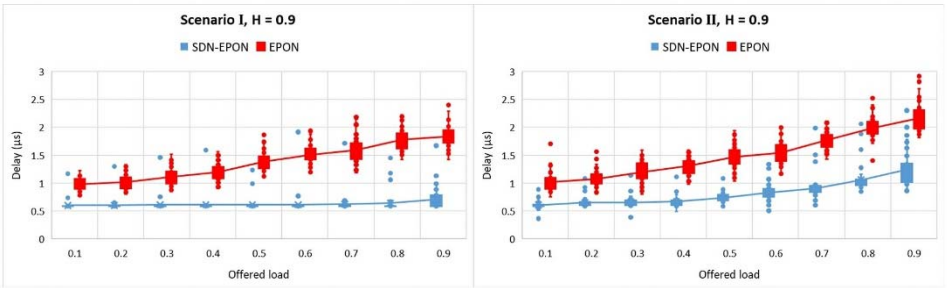

Figure 12 Queuing delay in SDN-EPON and EPON architectures under different number of end users (sixteen active ONUs on the left, eight active ONUs on the right), for different offered traffic load.

013 shows the throughput of both the downstream (left column) and upstream (right column) channels against the offered load for total ONUs, for CBR and self-similar traffic sources, where self-similarity is evaluated under sixteen and eight active ONUs (scenarios I and II, respectively). As shown in **¡Error! No se encuentra el origen de la referencia.**, the throughput is evaluated for the control and data frames traversed through path DE. As expected, the total throughput is very similar, although there is a small advantage for the SDN-based EPON architecture (3.7% better in the downstream, 2.9% better in the upstream), caused by a faster processing of packet is performed in existence of predefined rules, thus lowering the queueing delay. Results show the throughput for both architectures and scenarios (I and II) is quite similar no matter what configuration and Hurst parameters are used. As the number of packets is increased, the average transmission delay and the QoS will be degraded. Again, the traffic pattern does not have any relevant influence in the results.

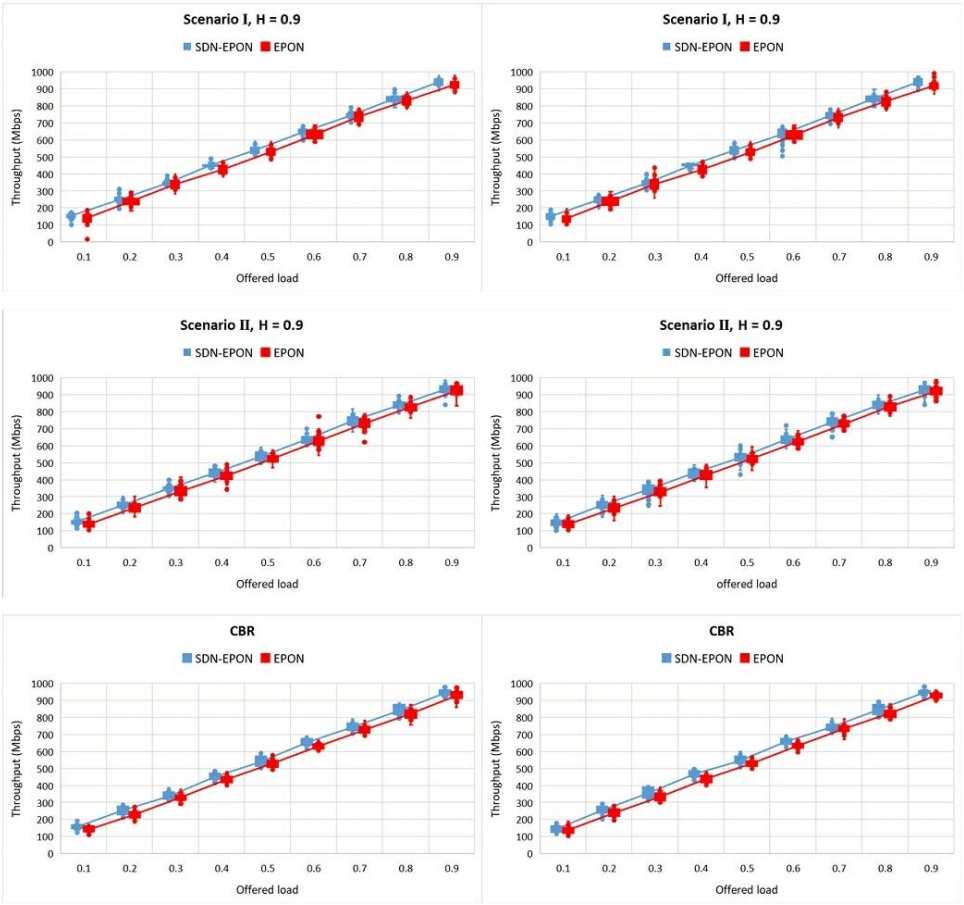

Figure 13 Downstream and upstream throughput in SDN-EPON and EPON architectures under different traffic sources and number of end users (sixteen active ONUs on the left, eight on the right), for different traffic patterns (CBR and self-similar) and different offered traffic load.

5. Conclusions

We have described a novel architectural design for SDN-controlled EPON networks, together with system implementation details. Our architecture minimizes the management and operational complexity of the EPON, while optimizing the flexibility and controllability of the network. An extension of MultiPoint MAC Control sub-layer is defined in the OLT to decouple some of its functionalities and distribute them between the SDN controller and the SDN OLT. The management and control functionalities of the MAC client and MAC control client are migrated to the SDN controller, (an example of the operations that take place at longer time scales), and the main functional blocks kept in the SDN-based OLT to be integrated with the OpenFlow switch (operations that work at short time scales are kept at the OLT). Synchronous operations are introduced in the OpenFlow architecture by defining a group of synchronous and non-synchronous ports (via operation of registers), and extending the OpenFlow's rules and messages. Several simulations have demonstrated a significant improvement in data packet delay and downstream and upstream throughput when compared with the legacy EPON architecture.

Among the advantages of our approach we want to emphasize that it opens a lot of possibilities related to the optimization in the resource usage. For example, we envision end-to-end QoS by the coordination of the SDN controllers in charge of the access, metro and core networks. Another important feature of our architecture is the possibility of sharing the same EPON infrastructure among different service providers, by providing separate slices of resources to each provider, under the control and coordination of the SDN controller. This opens the way for multitenant, virtualized EPON networks, with a substantial reduction in OPEX and CAPEX, and the creation of new business models in the field of optical access networks.

Our work is currently focused in the development of power-saving algorithms in the OLT and the ONUs, and its management by the SDN control plane. Power saving can be obtained by setting the laser transmitters and the receivers of both the OLT and ONUs to a sleep state when there is no traffic [28]. The SDN controller could modify the behaviour of the DBA and use the GATE/REPORT messages to switch the state of the ONU between sleep, doze, and active mode. The decision could be taken with the information transported in REPORT messages about the queue status of the ONU for the upstream channel, and the OLT queue information for the downstream channel. The action of setting off transmitters affects the traffic patterns by grouping the data frames in burst, thus introducing delays and jitter, and affecting the QoS. That is why a global approach in which the SDN controller has both the control over the QoS and the energy saving, and can reach an appropriate trade-off.

Acknowledgement

This work has been supported by the Ministerio de Economía y Competitividad of the Spanish Government under project TEC2016-76795-C6-1-R and AEI/FEDER.

References

[1] Glen Kramer, and Gerry Pesavento. Ethernet passive optical network (EPON): building a next-generation optical access network. IEEE Communications magazine. 2002;**40**:66-73.

[2] Ivica Cale, Aida Salihovic, and Matija Ivekovic. Gigabit passive optical network-GPON. In: 29th International Conference on Information Technology Interfaces; 25-28 June; Cavtat, Croatia. IEEE; 2007. p. 679-684.

[3] Glen Kramer, Lior Khermosh, Fumio Daido, Alan Brwon, Hosung Yoon, Ken-Ichi Suzuki, and Wang Bo. The IEEE 1904.1 standard: SIEPON architecture and model. IEEE Communications Magazine. 2012;**50**(9):98-108.
[4] IEEE standards Association. 1904.1-2013 - IEEE Standard for Service Interoperability in Ethernet Passive Optical Networks (SIEPON), 2013
[5] Ken-Ichi Suzuki. G.EPON and Current Status of Related Standardization. Jan. 2014 . https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr201401gls_s.html
[6] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig . Software-defined networking: A comprehensive survey. Proceedings of the IEEE. 2015;**103**(1):14-76.
[7] Nick McKeown, Tom Anderson, Hari Balakrishnan MIT, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review. 2008;**38**(2):69-74.
[8] Jane M Simmons. Optical network design and planning. Springer; 2014.
[9] Akhilesh S Thyagaturu, Anu Mercian, Michael P. McGarry, Martin Reisslein and Wolfgang Kellerer. Software defined optical networks (SDONs): A comprehensive survey. IEEE Communications Surveys & Tutorials. 2016;**18**(4):2738-2786.
[10] Lei Liu, Takehiro Tsuritani, Itsuro Morita, Hongxiang Guo, and Jian Wu. OpenFlow-based wavelength path control in transparent optical networks: A proof-of-concept demonstration. In: 37th European Conference and Exposition on Optical Communications; 2011; Geneva Switzerland. Optical Society of America; 2011. p. Tu--5.
[11] Lei Liu, Dongxu Zhang, Takehiro Tsuritani, Ricard Vilalta, Ramon Casellas, Linfeng Hong, Itsuro Morita, Hongxiang Guo, Jian Wu, Ricardo Martinez, and Raul Muñoz. First Field Trial of an OpenFlow-based Unified Control Plane for Multi-layer Multi-granularity Optical Networks. In: Optical Fiber Communication Conference; 4-8 March 2012; Los Angeles, California United States. Optical Society of America; 2012. p. PDP5D-2.
[12] Fernando N. N. Farias, João J. Salvatti, Eduardo C. Cerqueira, and Antônio J. G. Abelém. A proposal management of the legacy network environment using OpenFlow control plane. In: Network Operations and Management Symposium (NOMS); 16-20 April 2012; Maui, HI, USA. IEEE; 2012. p. 1143-1150.
[13] Pawel Parol, and Michal Pawlowski. Towards networks of the future: SDN paradigm introduction to PON networking for business applications. In: Computer Science and Information Systems (FedCSIS), Federated Conference on; 8-11 Sept. 2013; Krakow, Poland. IEEE; 2013.
[14] M.Ruffin, F.Slyne, C.Bluemm, N.Kitsuwan, S.McGettrick. Software Defined Networking for Next Generation Converged Metro-Access Networks. Optical Fiber Technolog. 2015;**26**:31-41.
[15] Hui Yang, Jie Zhang, Yongli Zhao, Jialin Wu, Yuefeng Ji, Yi Lin, Jianrui Han, and Young Lee. Experimental demonstration of remote unified control for OpenFlow-based software-defined optical access networks. Photonic Network Communications. 2016;**31**(3):568-577.
[16] Ahmed Amokrane, Jinho Hwang, Jin Xiao, and Nikos Anerousis. Software defined enterprise passive optical network. In: 10th International Conference on Network and Service Management (CNSM); 17-21 Nov. 2014; Rio de Janeiro, Brazil. IEEE; 2014. p. 206-411.
[17] Hamzeh Khalili, David Rincon, and Sebastia Sallent. Towards an Integrated SDN-NFV Architecture for EPON Networks. In: Meeting of the European Network of Universities and Companies in Information and Communication Engineering, EUNICE: Advances in Communication Networking ; Sept. 2014; Rennes France. Springer; 2014. p. 74-84.
[18] Chengjun Li, Wei Guo, Wei Wang, Weisheng Hu, and Ming Xia. Programmable bandwidth management in software-defined EPON architecture. Optics Communications. 2016;**370**:43-48.
[19] Steven S. W. Lee, Kuang-Yi Li, and Ming-Shu Wu. Design and Implementation of a GPON-Based Virtual OpenFlow-Enabled SDN Switch. Journal of Lightwave Technology. 2016;**34**(10):2552-2561.

[20] Larry Peterson, Ali Al-Shabibi, Tom Anshutz, Scott Baker, Andy Bavier, Saurav Das, Jonathan Hart, Guru Palukar, and William Snow . Central office re-architected as a data center. IEEE Communications Magazine. 2016;**54**(10):96-101.
[21] Open Networking Foundation. Open Network Operating System (ONOS) [Internet]. . Available from: http://onosproject.org/.
[22] Toshihiko Kusano. Architecture for an Access Network System Management Protocol Control under Heterogeneous Network Management [Internet]. 2014-10-23. Available from: http://www.patentsencyclopedia.com/app/20140314087
[23] Hamzeh Khalili, Sebastia Sallent, Jose Ramon Piney, and David Rincon. A proposal for an SDN-based SIEPON architecture. Optics Communications. 2017;**403**:9-21.
[24] G. Kramer, B. Mukherjee, and G. Pesavento. Interleaved Polling with Adaptive Cycle Time (IPACT): A Dynamic Bandwidth Distribution Scheme in an Optical Access Network. Photonic Network Communications, 4(1):89–107, 2002.
[25] Paola Garfias, Resource Management Research in Ethernet Passive Optical Networks (Ph.D thesis), Universitat Politecnica de Catalunya UPC-BarcelonaTech, 2013.
[26] David Rincón. Contributions to the Wavelet-based Characterization of Network Traffic. PhD Thesis, Universitat Politecnica de Catalunya - Barcelona, July 2007.
[27] David Rincón, Sebastià Sallent, Segmentation of fractal network traffic with wavelets and log-likelihood statistics, IEEE International Conference on Communications, ICC 2005, pp 11-15
[28] Jorge Ivan Argüello, Power saving in passive optical networks with distributed bandwidth allocation, MSc Thesis, Universitat Politecnica de Catalunya - Barcelona, July 2017.