Universidade de Aveiro Departamento de
Electrónica, Telecomunicações e Informática (DETI)
2016

**José Miguel
Duarte Maricato**

Redes de Acesso Definidas por Software

Software Defined Access Networks

Universidade de Aveiro Departamento de
Electrónica, Telecomunicações e Informática (DETI)
2016

José Miguel Duarte Maricato

Redes de Acesso Definidas por Software

Software Defined Access Networks

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Dr. António Teixeira e do Dr. Mário Lima, ambos do Departamento de Electrónica, Telecomunicações e Informática e do Instituto de Telecomunicações da Universidade de Aveiro.

o júri / the jury

presidente / president	**Prof. Doutor Nuno Borges de Carvalho** Professor catedrático da Universidade de Aveiro
vogais / examiners committee	**Professor Doutor António Luís Jesus Teixeira** Professor associado da Universidade de Aveiro (orientador)
	Prof. Doutor Pedro Tavares Pinho Professor adjunto do Instituto Superior de Engenharia de Lisboa

Agradecimentos

Gostaria desde já agradecer à Universidade de Aveiro, os anos aqui passados enriqueceram-me muito a nível pessoal e técnico, foi nesta instituição que conheci algumas das melhores pessoas que fazem parte da minha vida e onde vivi momentos que irão perdurar largos anos na minha memória.

Quero agradecer aos meus orientadores, Prof. Dr. António Teixeira e Prof. Dr. Mário Lima pela motivação dada e confiança depositada em mim, que me ajudaram a ultrapassar as dificuldades e a melhorar o meu trabalho.

Aproveito esta oportunidade para agradecer aos meus pais, irmão, família e namorada, por acreditarem nas minhas capacidades e pela paciência que tiveram comigo, quer nos momentos altos quer nos baixos, a fé incondicional depositada em mim motivou-me para finalizar este projeto.

A todos os meus amigos dirijo um grande e sincero agradecimento, pela amizade, apoio e compreensão ao longo destes anos.

Acknowledgements

I want to thank the University of Aveiro, these past years have enriched me at a personal and technical level, in this institution I met some of the best people who are now part of my life and where I lived moments that will endure many years in my memory.

I want to thank my supervisors, Prof. Dr. António Teixeira e Prof. Dr. Mário Lima for the given motivation and trust in me, which helped me overcome the difficulties and improve my work.

I take this opportunity to thank my parents, brother, family and girlfriend, for believing in my capabilities and for their patience with me, both in the highs or the lows, the unconditional faith placed in me motivated me to finalize this project.

To all my friends I give a big and sincere thanks, for the friendship, support and understanding over the years.

Palavras Chave

PON, G-PON, NG-PON2, TWDM-PON, OMCI, PLOAM, Virtualização de Redes, SDN, NFV

Resumo

Com o crescimento da utilização da Internet e o consumo de largura de banda a crescer exponencialmente devido ao crescente número de utilizadores de equipamentos de nova geração e à criação de novos serviços que consomem cada vez maiores larguras de banda, é necessário encontrar soluções para satisfazer estes novos requisitos. As redes ópticas passivas (PON) prometem solucionar esses problemas, oferecendo um melhor serviço aos utilizadores e provedores. As redes PON são muito atrativas pois não dependem de elementos ativos entre os seus pontos terminais, resultando em baixos custos de manutenção e uma maior eficiência de operações.

As tecnologias PON abordadas nesta dissertação são o G-PON (*Gigabit PON*), actualmente padronizada e implementada nas redes de accesso pelo mundo, e o NG-PON2 (*Next-Generation PON 2*), que será o próximo passo na evolução das redes de acesso e que atualmente se encontra em processo de estudo e padronização. O NG-PON2 deve co-existir na mesma rede de distribuição ótica do G-PON, de forma a re-utilizar as infrastruturas já construidas e consequentemente proteger o investimento inicial dos provedores.

As redes definidas por software (SDN) é uma arquitetura emergente que desassocia o controlo da rede e funções de encaminhamento do hardware a que pertencem, possibilitando a que o controlo da rede seja programável, permitindo a implementação de soluções capazes de resolver o problema do aumento da complexidade das redes e criação de serviços inovadores. O principal foco de estudo será nas SDN como mecanismo de virtualização dos elementos da rede.

Nesta dissertação é estudado as arquiteturas do G-PON e NG-PON2 no contexto das recomendações do ITU-T G.984.x e G.989.x respetivamente, e o estudo da tecnologia SDN através da documentação disponível online. Com base nos estudos efetuados irá ser sugerido uma arquitetura de um servidor que permite o controlo de elementos da infrastrutura G-PON e NG-PON2, intoduzindo os conceitos das SDN e virtualização na rede de acesso.

Keywords PON, G-PON, NG-PON2, TWDM-PON, OMCI, PLOAM, Network Virtualization, SDN, NFV

Abstract

With the increase of internet usage and the exponential growth of bandwidth consumption due to the increasing number of users of new generation equipments and the creation of new services that consume increasingly higher bandwidths, it's necessary to find solutions to meet these new requirements. Passive optical networks (PONs) promise to solve these problems by providing a better service to users and providers. PON networks are very attractive since they don't depend on active elements between their end points, leading to lower maintenance costs and better operational efficiency.

PON technologies addressed in this dissertation are the G-PON (Gigabit PON), currently standardized and implemented in access networks across the world, and the NG-PON2 (Next-Generation PON 2), which is the next step on access networks evolution and is currently on the process of study and standardization. The NG-PON2 must co-exist on the same optical distribution network of the G-PON, so it re-utilizes the already built infrastructures and consequently protect providers initial investment.

Software Defined Networks (SDN) is an emerging architecture that decouples network control and forwarding functions from the hardware they belong, making possible for network control to be programmable, enabling the implementation of solutions capable of solving the increasing complexity of the networks problem and the creation of innovative services. The study main focus is the SDN as an enabling mechanism for network elements virtualization.

In this dissertation is studied the G-PON and NG-PON2 architectures in the context of the ITU-T G.984.x and G.989.x recommendations respectively, and the study of the SDN technology through documentation available online. And based on the studies made it's going to be proposed a server architecture that enables the control of G-PON and NG-PON2 infrastructure elements, introducing virtualization SDN concepts on access networks.

Contents

List of Figures

List of Tables

List of Acronyms

ACK	Acknowledge
Alloc-ID	Allocation Identifier
API	Application Programmatic Interface
ARP	Address Resolution Protocol
AON	Active Optical Network
BER	Bit-error Rate
BIP	Bit Interleaved Parity
br-ex	External Bridge
br-int	Integration Bridge
br-tun	Tunneling Bridge
CAPEX	Capital Expenditure
CO	Central Office
CRC	Cyclic Redundancy Check
DBA	Dynamic Bandwidth Assignment
DBRu	Dynamic Bandwidth Report upstream
DG	Dying Gasp
DNAT	Destination NAT
EMS	Element Management System
ETH	Ethernet
FEC	Forward Error Correction
FS	Framing Sublayer
FTTB	Fiber-to-the-Building
FTTC	Fiber-to-the-Cabinet
FTTCell	Fiber-to-the-Cell
FTTH	Fiber-to-the-Home
FTTO	Fiber-to-the-Office
G-PON	Gigabit PON
GEM	G-PON Encapsulation Method
GTC	G-PON Transmission Convergence
HEC	Header Error Control
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector
L2	OSI Model Layer 2

L3	OSI Model Layer 3
L4	OSI Model Layer 4
Logic-ID	Logic Identifier
MAC	Media Access Control
MDU	Multi Dwelling Unit
MIB	Management Information Base
ML2	Modular Layer 2
MTU	Multi-Tenant Unit
NAT	Network Address Translation
NFV	Network Functions Virtualization
NFV MANO	NFV Management and Orchestration
NFVI	NFV Infrastructure
NG-PON2	Next-Generation PON 2
NIC	Network Interface Controller
NRZ	Non-return-to-zero
OAM	Operations, Administration and Maintenance
ODN	Optical Distribution Network
OLT	Optical Line Termination
OMCC	ONU Management and Control Channel
OMCI	ONU Management and Control Interface
ONF	Open Networking Foundation
ONT	Optical Network Termination
ONU	Optical Network Unit
ONU-ID	ONU Identifier
OPEX	Operational Expenditure
OSS	Operations Support System
OvS	Open vSwitch
OvSDB	OvS Database
PCBd	Physical Control Block downstream
PHY	Physical Interface
PLI	Payload Length Indication
PLOAM	Physical Layer Operations, Administration and Maintenance
PLOAMd	PLOAM downstream
PLOAMu	PLOAM upstream
PMD	Physical Media Dependent
PON	Passive Optical Network
Port-ID	Port Identifier
POTS	Plain Old Telephone Service
PSync	Physical Synchronization
QoS	Quality of Service
REST	Representational State Transfer
RF	Radio Frequency
RG	Residential Gateway
S-VLAN	Service V-LAN

SDN	Software Defined Networks
SDU	Service Data Unit
SNAT	Source NAT
T-CONT	Transmission Container
TC	Transmission Convergence
TCP	Transport Control Protocol
TDM	Time Division Multiplexing
TDMA	Time Division Multiple Access
TLS	Transport Layer Security
TWDM-PON	Time and Wavelength Division Multiplexing PON
UDP	User Datagram Protocol
UNI	User Network Interface
US BWmap	Upstream Bandwidth Mapping
VI	Virtualized Infrastructure
VIF	Virtual Interface
VLAN	Virtual Local Area Network
VM	Virtual Machine
VNF	Virtual Network Function
vOLT	Virtualized Optical Line Termination
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing
xDSL	Digital Subscriber Line
XG-PON	10-Gigabit PON
XGEM	XG-PON Encapsulation Method
XGTC	XG-PON Transmission Convergence

Chapter 1

Introduction

1.1 Motivation

Nowadays there's a huge number of equipments that are connected to the network (e.g. laptops, notebooks, smartphones, gaming consoles), and this number is rising every day, and all of them allows access to a set of services like email, internet, online gaming, live streaming, ultra High Definition, big file transfers, backups, and many other services, all of them competing for the same resources. The exponential growth of network usage, as seen in figure 1.1, makes the infrastructure behind more complex, even with the fact that these networks are frequently improved and automated, the need for human administrators is rising, and with the rapid growth of the networks, the administrators aren't able to manage this growth quickly enough.

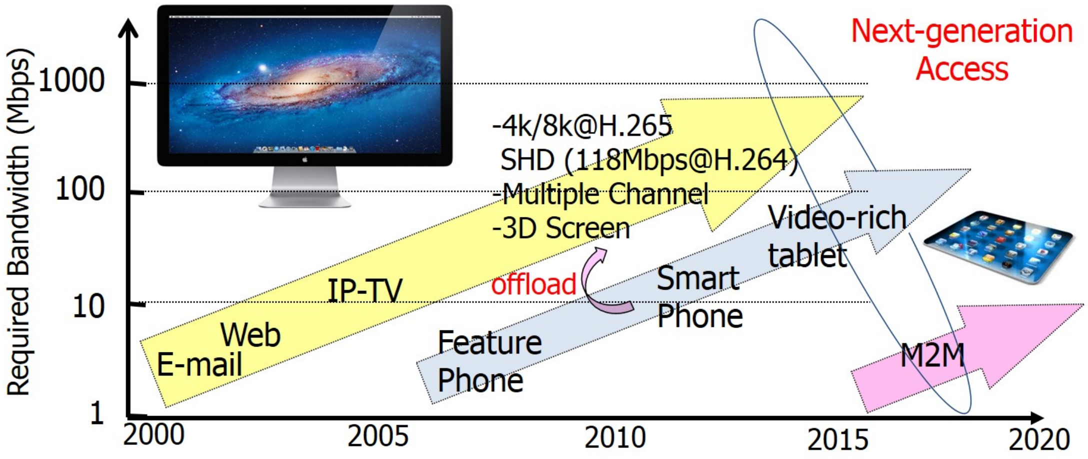


Figure 1.1: Future needs of bandwidth [1]

Meeting with current market requirements is virtually impossible with current Gigabit PON (G-PON) and Ethernet PON (E-PON) network architectures. Faced with flat or reduced budgets, enterprise Information and Technology (IT) departments are trying to squeeze the most from their networks, using device-level management tools and manual processes. Carriers face similar challenges as demand for mobility and bandwidth increases, profits are being eroded by escalating capital equipment costs and flat or declining revenue. Existing network architectures weren't designed to meet the requirements of today's customers and enterprises, which leads to network providers being constrained by the limitations of current networks.

Also, due to business models and needs, the networks are constantly growing and being reconfigured. New services, new applications, bigger and bigger data centres, and other business needs, add complexity to the network and raise their operational and implementation costs. The aggregation network is an example of this complexity, where the traffic from all service providers is aggregated and connected to the end customers using hierarchical switches and routers linked to each other via Ethernet (ETH) or fiber cables, so they can provide reliability, and the amount of equipments needed rise with the number of services offered, and it isn't an easy task to configure and manage all of them.

A requirement that is still missing is the possibility of a single and simple network management interface that can provide a global view of the entire network, and gather information like data flow, overloaded connections, and more. Software Defined Networks (SDN) will be an upgrade for network infrastructures, where the capability to offer a programmable network provides a Quality of Service (QoS) via dynamic and reconfigurable traffic separation. With these tools the administrators can have an idea of how the network is being used and ensure that its resources are reserved and available to its needs. Services like healthcare and medicine, who are starting to use long distance clinical healthcare to patients or between facilities, require priority over other services and a high quality of multimedia transmission, and SDN can work as the tool to ensure the QoS needed to offer this services.

As seen, networks have become a critical part of society infrastructure. With the rising of network usage they have become complex, difficult to optimize, expensive, static and a barrier for new services to thrive. This design no longer makes sense, the dynamic computing and storage needs of virtualized data centres demands the transformation of the networks, because today's network requires scalability, programmability, agility and automated management. This mismatch between market requirements and network capabilities has brought the industry to a tipping point. The Next-Generation PON 2 (NG-PON2) recommendations are being developed in response to increasing bandwidths, and SDN architectures capabilities are being tested to simplify management and maximize benefits of access networks.

1.2 Objectives

The purposes of this dissertation are:

- Study of the optical fiber access networks G-PON and NG-PON2 according to their International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) recommendations, analysing the communication structure between the Optical Line Termination (OLT) and the Optical Network Units (ONUs).

- Study the G-PON and NG-PON2 management mechanisms provided on their recommendations and available literature.
- Observe and derive the conditions and technologies for SDN, focusing mainly on the study of the virtualization trends.
- Apply the considerations studied on SDN in the context of the G-PON and NG-PON2 recommendations.

1.3 Structure

This dissertation is organized in five chapter:

- Introduction
- State of Passive Optical Network (PON) Technologies
- Software Defined Networks (SDN) and Network Functions Virtualization (NFV)
- Software Defined Access Networks
- Conclusions

In the first chapter is presented the evolution and tendencies of network usage, giving the motivation for merging SDN and Network Functions Virtualization (NFV) technologies with G-PON and NG-PON2 access networks.

In the second chapter is shown the different types of the telecommunication network structure. The G-PON and NG-PON2 recommendations are studied, and their architecture, Transmission Convergence (TC) layer and management mechanisms are derived.

The third chapter analyses the SDN and NFV architectures, pointing the benefits introduced in the networks and the different implementation classes. Furthermore, the OpenFlow and OpenStack technologies used by SDN and NFV are studied.

The fourth chapter proposes a controller server architecture used for running virtualized functions of network equipments presented in the access network. Also a Virtualized Optical Line Termination (vOLT) implementation is proposed, capable of providing a programmable, multi-tenant and remotely controlled OLT control plane.

The last chapter presents the conclusions obtained through the research done on the topic. It's also explained some future work necessary to enable a vOLT implementation.

1.4 Contributions

The contributions of this dissertation are:

- Assessment of the G-PON and NG-PON2 management features that can be virtualized.
- Proposal of a controller server architecture relying on NFV technology and capable of supporting different types of virtualized network functions.
- Proposal of a vOLT using the controller server architecture. The vOLT architecture, capabilities and function methodology are delineated.

Chapter 2

State of Passive Optical Networks (PON) Technologies

2.1 Introduction

With the standardization of Passive Optical Network (PON) technologies due to the necessity to lower the costs of optical access networks, the need for evolution and the creation of new revenue models in the telecommunications market, is making the implementation of PONs intensively rising all over the world.

The current standardised technologies G-PON and 10-Gigabit PON (XG-PON) don't meet the rising necessity of bandwidth, so there's a need for evolution, and to meet this there's the NG-PON2 technology. In this way, and to maintain the Operational Expenditure (OPEX) reduction and to protect the initial investment, operators should keep the current wavelength planning, so there is a need for co-existence in the same fiber of the current G-PON and the future access networks.

The ITU-T is an organisation dedicated to the standardization of radio and telecommunications wavelengths, along with Full Service Access Network (FSAN) working group, founded by seven global operators in 1995, show great activity in the study of these networks. According to them, NG-PON2 belongs to the next generation of PONs, and is currently being standardised and is expected to being implemented in the near future.

In this chapter there is a brief description of the importance of PON networks, and the explanation of the current G-PON and NG-PON2 technologies according to their ITU-T recommendations.

2.2 Telecommunications Network

2.2.1 Network Types

The current telecommunications network structure consists of three sub-networks: backbone or core network, metro or regional network and the access network. Figure 2.1 shows a simple scheme of a telecommunications network architecture.

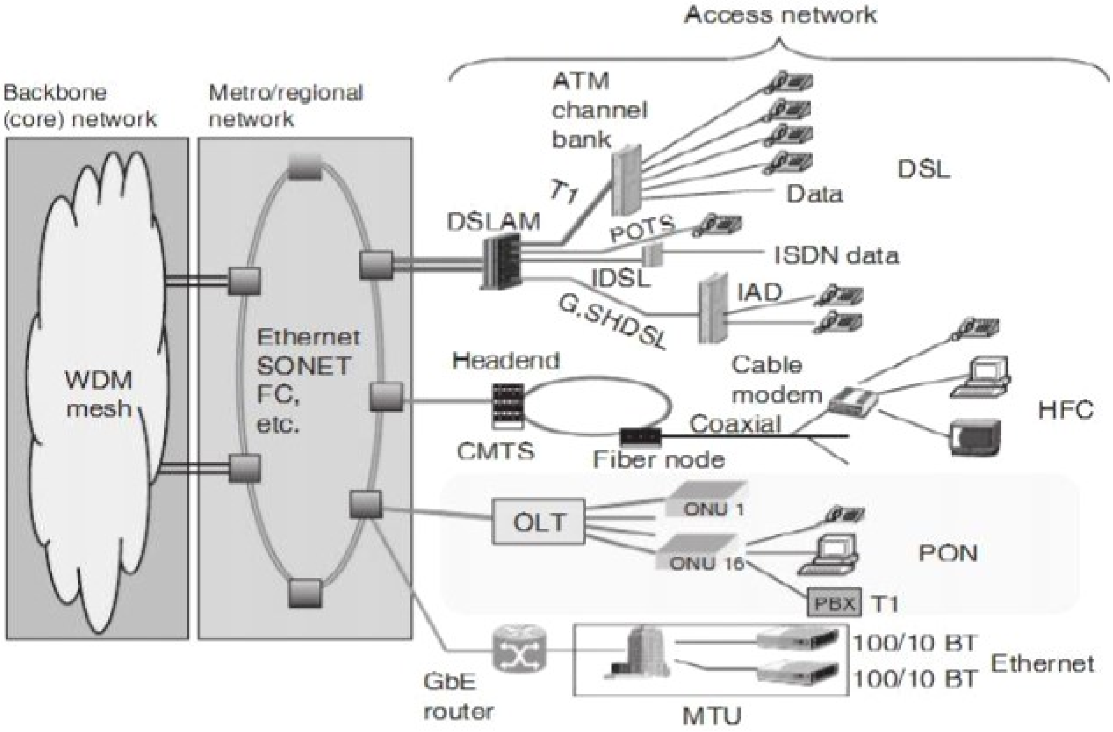


Figure 2.1: Architecture of a telecommunication network [2]

The core network interconnects all the metro networks of a country. The metro networks aggregate high tributary traffic from the Central Offices (COs), pass traffic addressed to other metro networks to the core network and delivers to the respective COs the remaining traffic.

The structures of core and metro networks are usually more uniform than access networks and their costs are shared among large numbers of network providers. Finally, the access network provides end-user connectivity, this is, connects service providers with their customers.

2.2.2 Access Network

The access network is the last segment of the telecommunications network that runs from the service providers facility to the home or business customers. The most cost efficient solution in access network that accommodates high bandwidth services, for example, Video on Demand (VoD), High Definition Television (HDTV), and various future forms of videos, is the PON. PON architectures must be simple and easy to operate, the network in itself doesn't have any switching and doesn't need to be controlled. Furthermore, passive components don't need to be powered, except at the terminations, which provide efficient cost savings to service providers. [3]

PONs are constituted of an OLT located in the CO, which connects to the remote node, this network is called the feeder network. The remote nodes are connected to a set of ONUs that terminate the fiber and reach the customers, called distribution network, as can be seen in figure 2.2.

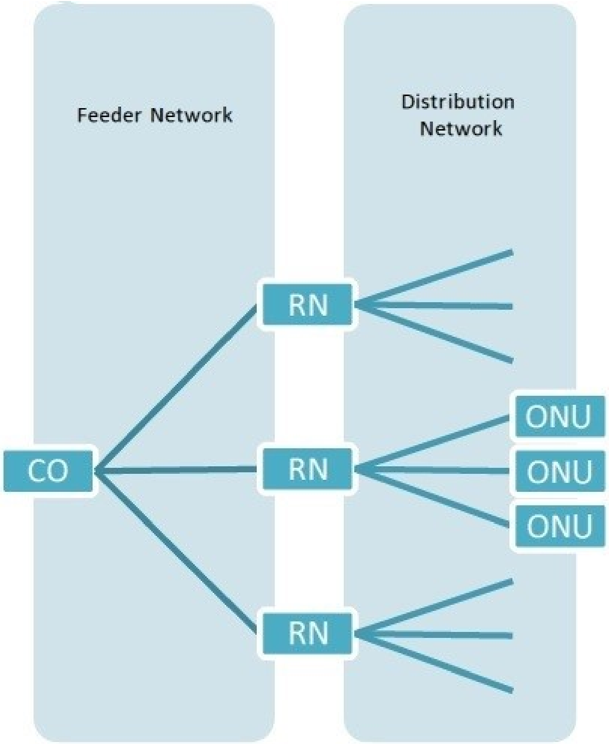


Figure 2.2: High-Level architecture of an access network

Distribution Network

PONs have several types of fiber distribution architectures. Optical fiber is used in both feeder and distribution networks, and the type of fiber deployment depends on the ONU location and fiber length. Three of the most common extents for fiber deployment in access are: fiber-to-the-home/building/cabinet (FTTH/B/C). Its designation depends on where the fiber is terminated, being the remain distribution done by copper loops wire or ETH equipment, when copper loop isn't used and the end user connects directly to the fiber it's a FTTH deployment. Example in figure 2.3.

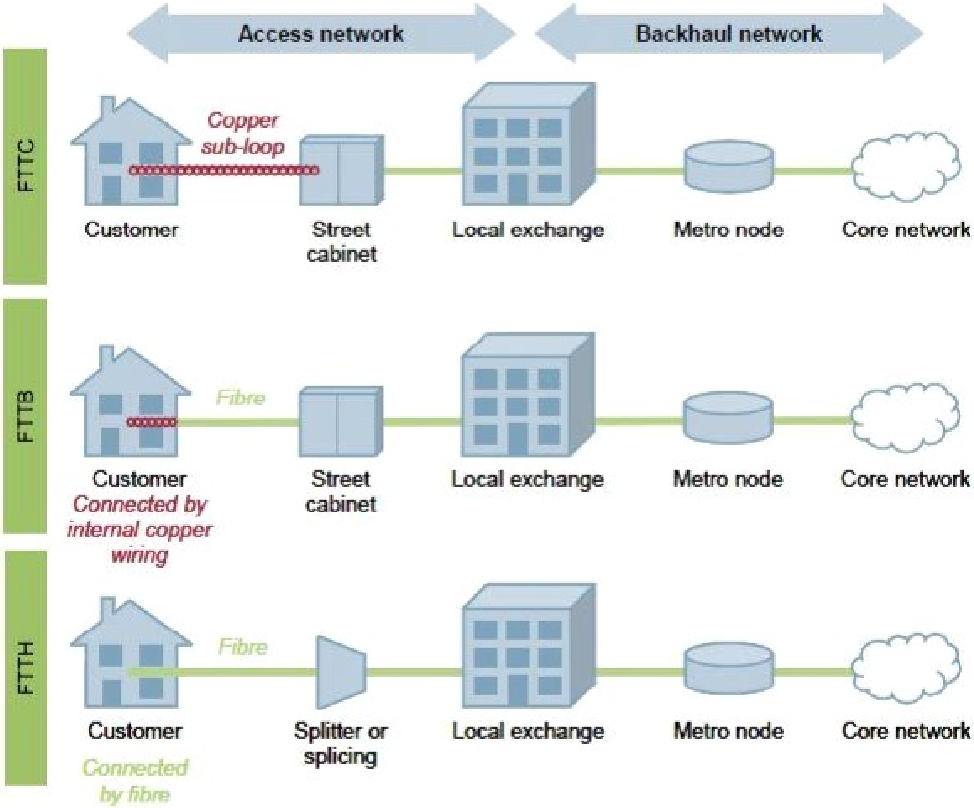


Figure 2.3: Fiber distribution architectures [4]

Next generation access networks are characterised by attempting to shorten or remove the copper loop, that is why FTTH are receiving a special attention because of the knowledge that connecting homes directly to the optical fiber cable can enable enormous improvements on the ever-increasing demand for bandwidth from customers. [4]

2.2.3 Passive Optical Network (PON)

For service providers, one of the most important decision to make is the acquisition of network equipments, and there should be a balance between the implementation costs of these equipments and the revenue they bring from the QoS and bandwidth they provide to the network.

The remote node of a network can be either passive or active. A fiber distribution architecture that makes use of an active remote node is an Active Optical Network (AON), and this one requires constant power supply, backup power and a cabinet for its placement, raising Capital Expenditure (CAPEX) and OPEX. While a fiber distribution architecture that uses a passive remote node is referred as a PON, and do to its passive elements it doesn't require the power supply, backup power and cabinet as the AON. This means that in this access networks, its deployment involves lower CAPEX and its maintenance lower

OPEX than the AONs since there aren't any active elements between the CO and the customer, only in these ends there are present some active components.

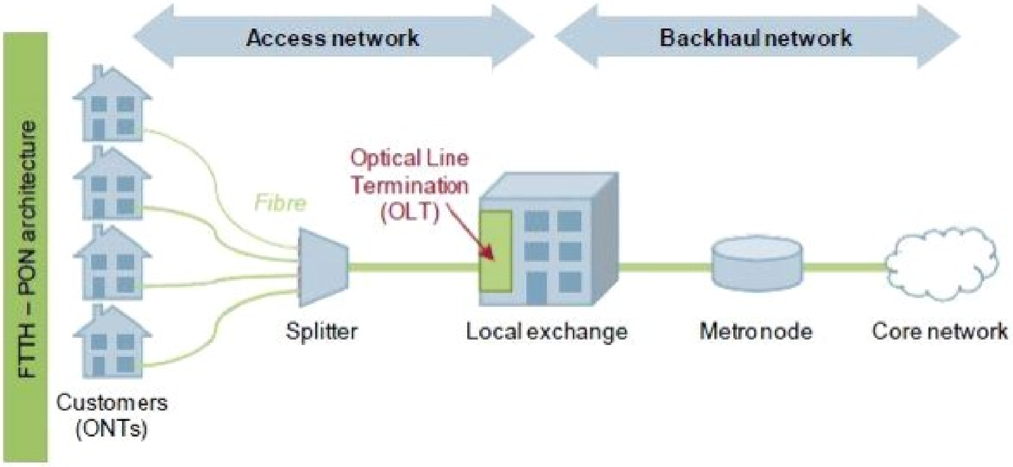


Figure 2.4: PON architecture [4]

From the CO OLT, a single-mode optical fiber is connected to the entrance of a 1:N passive optical splitter, and the N exits of the splitter are connected to the customers' ONUs through individual single-mode optical fibers.

This set of single-mode optical fibers and passive optical components that are installed between the CO and the customers' residences is called the Optical Distribution Network (ODN). Furthermore, a PON is also characterized for using a Wavelength Division Multiplexing (WDM) that enables the multiplexing of multiple optical carriers in a single optical fiber using different wavelengths for each carrier. WDM also enables a bidirectional (downstream and upstream) data flow through a single fiber.

PON main advantages are the fact that operators can greatly reduce power consumption outside the CO, since they don't need to install or maintain active network components, such as a Digital Subscriber Line Access Multiplexer (DSLAM) or an ETH switch, and also the sharing of a single optical interface in the OLT between multiple customers, reducing the requisition of space in the CO. This advantages leads to less CAPEX and OPEX and a greater reliability of the network, making PONs the most widespread implemented technology. [4]

2.3 Gigabit-capable Passive Optical Network (G-PON)

2.3.1 Architecture

G-PON is defined by the ITU-T in a series of G.984.x (x = 1, 2, 3, 4, 5, 6) recommendations that define general characteristics of G-PON systems and the specifications from the Physical Media Dependent (PMD) and the TC layer.

The typical architecture of G-PON follows the PON model. This means that the only active elements are the OLT and ONUs. Starting in the CO, a single-mode optical fiber is connected to a passive optical splitter near the customers' location, and at this point, the splitter divides the optical power into several different paths, each of them connected to the customers' ONUs. Furthermore, a WDM coupler is used for multiplexing the data/voice carrier and the video carrier in a single fiber by using different wavelengths, and also allows the bidirectional data flow in the fiber, as presented in figure 2.5.

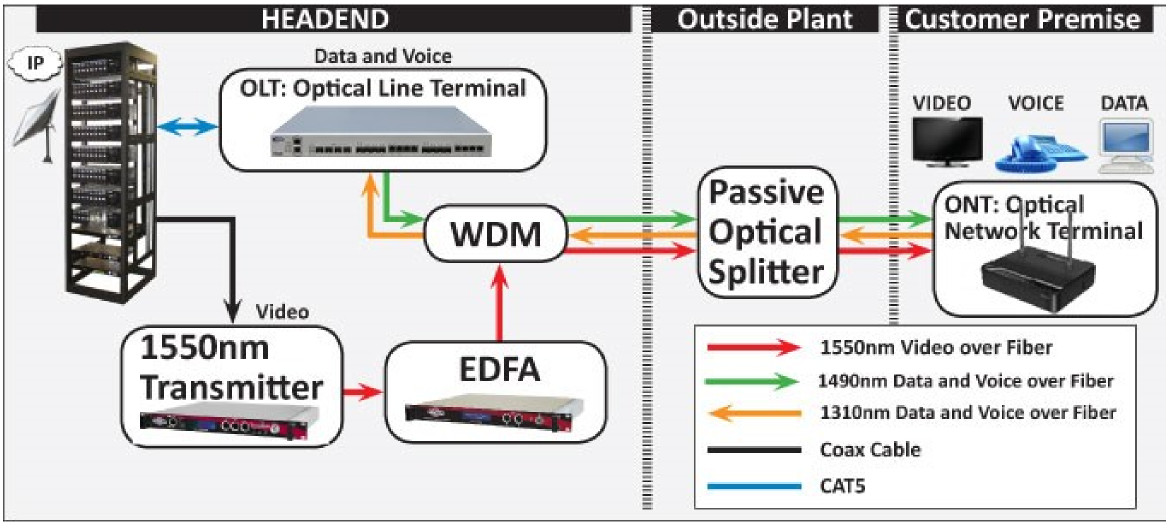


Figure 2.5: G-PON architecture schematic [5]

	FTTH	FTTC	FTTB	
			Business	MDU
Asymmetric broadband services	✓	✓	✗	✓
Symmetric broadband services	✓	✓	✓	✓
POTS and ISDN	✓	✓	✓	✓
Private line services	✗	✗	✓	✗
xDSL backhaul	✗	✓	✗	✗

Table 2.1: Different services supported by each G-PON architecture [6]

The common architectures for G-PON are the same as in figure 2.3, and they vary depending on the ONU location. The differences between them are mainly due to the different services supported by each one of them, shown in table 2.1.

2.3.2 Characteristics

Bit Rate

G-PON defines a different series of downstream and upstream nominal bit rates, these are shown in table 2.2.

Transmission direction	**Nominal bit rate**
Downstream	1.25 Gbit/s
	2.5 Gbit/s
Upstream	0.155 Gbit/s
	0.622 Gbit/s
	1.25 Gbit/s
	2.5 Gbit/s

Table 2.2: G-PON nominal bit rate [7]

The target standardized systems have nominal line rates pairs (downstream/upstream) of: [7]

- 1.25 Gbit/s | 0.155 Gbit/s
- 1.25 Gbit/s | 0.622 Gbit/s
- 1.25 Gbit/s | 1.255 Gbit/s
- 2.5 Gbit/s | 0.155 Gbit/s
- 2.5 Gbit/s | 0.622 Gbit/s
- 2.5 Gbit/s | 1.25 Gbit/s
- 2.5 Gbit/s | 2.5 Gbit/s

Line Coding

G-PON uses Non-return-to-zero (NRZ) line coding in both upstream and downstream directions. [7]

Physical Reach

Physical reach is the maximum physical distance between the ONU and the OLT. In G-PON, two options are defined for the physical reach: 10 km and 20 km. It's assumed that 10 km is the maximum distance over which a Fabry-Perot Laser Diode can be used in the ONU for high bit rates such as 1.25 Gbit/s or above. [6]

Split Ratio

In G-PON, split ratios of up to 1:64 are commonly used, and with modern optical modules, split ratios can go up to 1:128.

Network operators expect to have the maximum possible customers using their access networks, so the larger the split ratio, the more attractive it's to them. However, larger split ratios imply greater optical splitting, creating the need for increased power budget to support the physical reach. [6]

Wavelength Allocation

G-PON has two systems of operating wavelength, depending on the usage of one or two fibers. In the case of only one fiber, it's defined that the operating wavelength range for the downstream direction is 1480-1500 nm and for the upload direction is 1260-1360 nm. In the case of two fibers, one for each transmission direction, the operating wavelength range for each one is 1260-1360 nm. [7]

The typical architecture of G-PON systems must also enable the transmission, in the same fiber, of video signal, and the allocated bandwidth for the service is 1550-1560 nm in the downstream direction. [8]

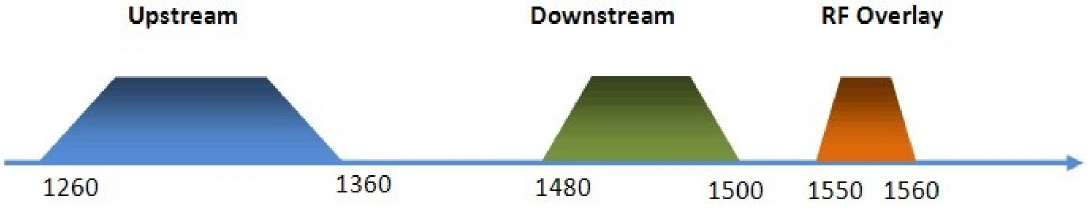


Figure 2.6: G-PON bandwidth allocation with one fiber [9]

Forward Error Correction (FEC)

In G-PON, Forward Error Correction (FEC) is used by the transport layer in the downstream direction, and is based on transmitting the data in an encoded format. The encoding introduces redundancy, which allows the decoder to detect and correct the transmission errors. By using FEC, transmissions with a low error rate are achieved, avoiding re-transmissions.

FEC results in an increased link budget by approximately 3-4 dB. Therefore, a higher bit rate and longer distance from the OLT to the ONUs can be supported, as well as a higher number of splits per single PON tree. [10]

2.3.3 G-PON Transmission Convergence (GTC)

G-PON Transmission Convergence (GTC) layer is based on Recommendation ITU-T G.984.3 [10].

G-PON Encapsulation Method (GEM)

G-PON Encapsulation Method (GEM) is identified as the sole data transport scheme in the TC layer, and it provides a connection-oriented, variable-length framing mechanism for the transport of Service Data Units (SDUs) over the PON and is independent of the type of the Service Node Interface (SNI) at the OLT, as well the type of User Network Interfaces (UNIs) at the ONUs.

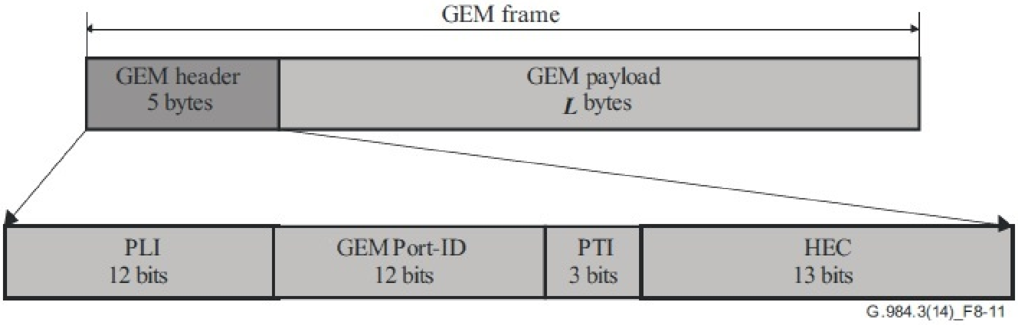


Figure 2.7: GEM frame format [10]

In figure 2.7 is shown the GEM frame, containing a 5 bytes GEM header and a variable size GEM payload field. In the GEM header there is a Payload Length Indication (PLI), GEM Port Identifier (Port-ID), Payload Type Indicator (PTI) and Header Error Control (HEC) fields. The GEM payload contains the GEM encapsulation of an ONU Management and Control Interface (OMCI) message or ETH frame. An ETH frame comprises the destinations address, source address, Etype/size, user data and Frame Check Sum (FCS). [10]

Field	Functionality
PLI	Length in bytes of the SDU contained in the GEM frame payload field
GEM Port-ID	Unique traffic identification on the PON in order to provide traffic multiplexing
PTI	Indicates the content type of the payload and its appropriate treatment
HEC	Used for error detection and correction

Table 2.3: GEM header fields [10]

Downstream Transmission

In the G-PON downstream direction, data packets are transmitted in a broadcast manner. Frames of 125μs interval with well-defined boundaries and fixed are sent from the OLT to the ONUs, all ONUs receive the same frames. A GEM Port-ID is assigned to each ONU by the OLT, and each GEM Port-ID is unique per PON interface. At the transmission, the OLT embeds a GEM Port-ID as a key to identify the frames that belong to different ONUs, and each ONU filters the downstream frames based on their GEM Port-IDs and processes only the frames that belong to that ONU. [10, 11]

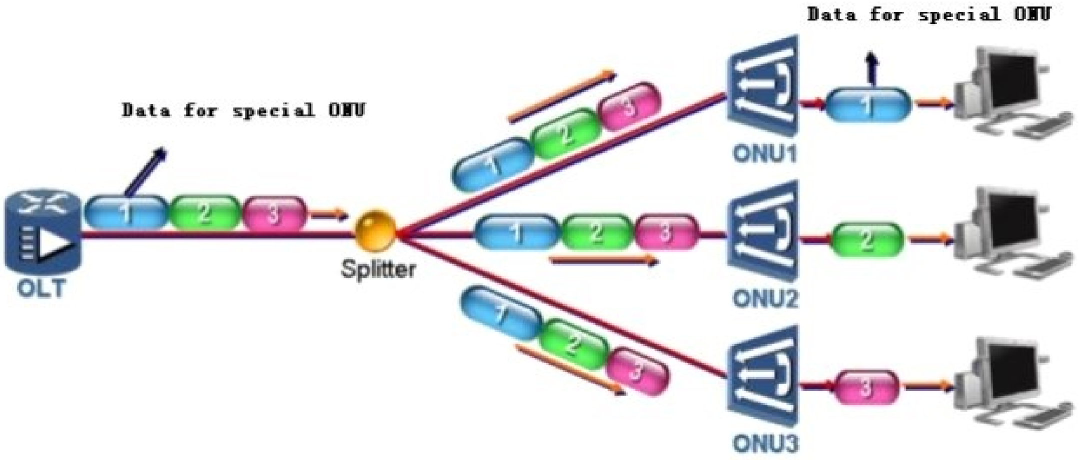


Figure 2.8: Downstream data transmission [11]

Downstream Frame

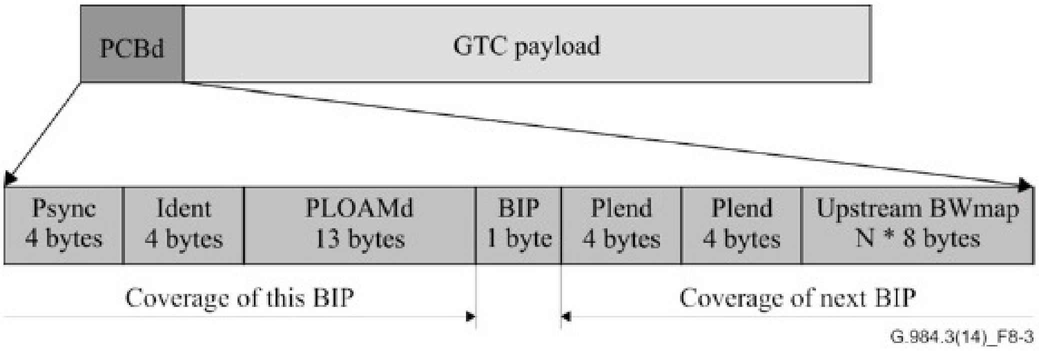


Figure 2.9: Downstream GTC frame [10]

The downstream GTC frame contains the Physical Control Block downstream (PCBd) header and the GTC payload that has the GEM frames. The ONUs read the PCBd header of the frames to get the related information, and its own payload part according to the Port-ID field of the GEM frames. In figure 2.9 is shown the downstream frame. The

PCBd header contains the Physical Synchronization (PSync), Ident, PLOAM downstream (PLOAMd), Bit Interleaved Parity (BIP), two Payload Length (PLend) and Upstream Bandwidth Mapping (US BWmap) fields. [10]

Field	Functionality
PSync	States the beggining of the downstream frame
Ident	Used to provide error tolerance, including superframe counter and FEC
PLOAMd	Contains the Physical Layer Operations, Administration and Maintenance (PLOAM) message
BIP	Used to check link errors by bit interleaved parity
Plend	Specifies the length of the US BWmap field
Plend	Sent twice for error robustness
US BWmap	Indicates the upstream bandwidth allocation map for the upstream grants of the Allocation Identifiers (Alloc-IDs)

Table 2.4: PCBd header fields [10]

Upstream Transmission

In the G-PON upstream direction, to avoid collision between data packets, is used a Time Division Multiple Access (TDMA) protocol. The OLT grants upstream transmission opportunities to the traffic-bearing entities within the ONUs. The ONUs traffic-bearing entities have a US BWmap field that specifies the recipient, that are identified by their Alloc-IDs, and the time interval for a particular bandwidth allocation. Using the TDMA protocol, the bandwidth allocations to different Alloc-IDs are multiplexed in time as specified by the OLT in the US BWmaps transmitted in the downstream direction. Within each bandwidth allocation, the ONU uses the GEM Port-ID as a multiplexing key to identify the GEM frames that belong to different upstream logical connections. [10, 11]

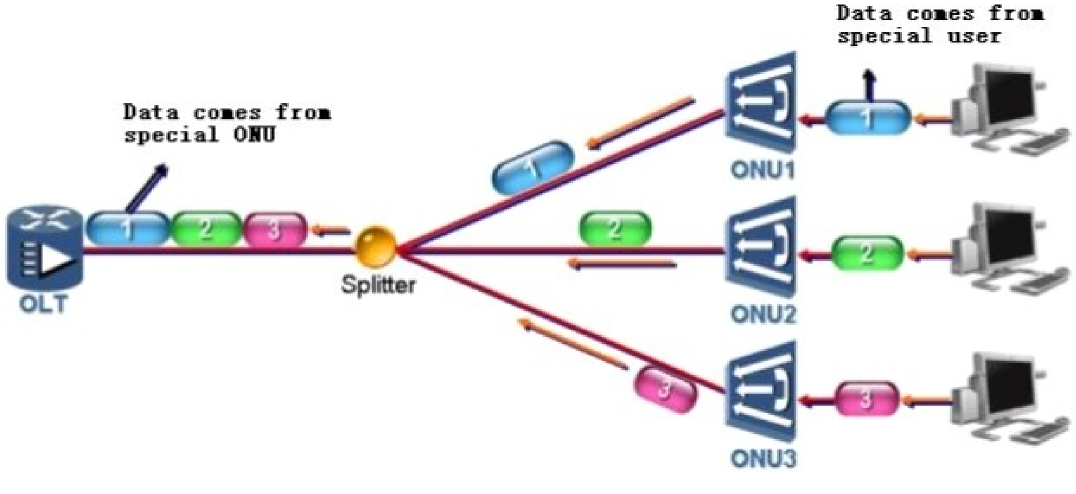


Figure 2.10: Upstream data transmission [11]

Upstream Frame

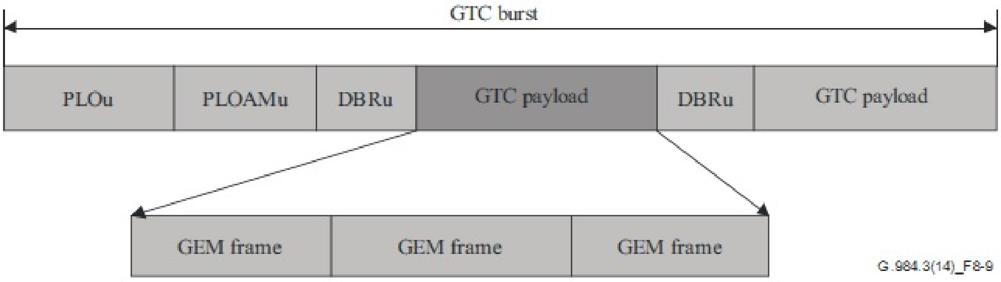


Figure 2.11: Upstream GTC burst [10]

The upstream GTC burst, in figure 2.11, contains one or more transmission units. Every unit has a Physical Layer Overhead upstream (PLOu) and GTC payload fields, the PLOAM upstream (PLOAMu), Power Levelling Sequence upstream (PLSu) and Dynamic Bandwidth Report upstream (DBRu) fields are only transmitted when needed. When the OLT receives the upstream GTC burst it unpacks it and reads the ONU Identifier (ONU-ID) to know the sender ONU, analyses the PLOAMu field and passes it to the PLOAM client to apply the corresponding action, the PLSu field is analysed and the power mode is changed if needed, then allocates an upstream bandwidth grant dynamically according to the DBRu, and finally passes the GTC payload to the corresponding receiver. [10]

Field	Functionality
PLOu	Has the information regarding the ONU-ID so the OLT knows the source of the frame, and also informs if a PLOAMu message is waiting to be sent
PLOAMu	Contains the PLOAM message
PLSu	Used for power measurements of the ONU
DBRu	Contains the traffic status regarding the Transmission Containers (T-CONTs) of the ONU if the Dynamic Bandwidth Assignment (DBA) mode is running
GTC payload	Contains the upstream GEM frames

Table 2.5: Upstream GTC burst fields [10]

Ranging

Since the ONU sends information to the OLT using TDMA, each ONU must be precisely synchronized with all others ONUs. To achieve synchronization across all of them, the OLT first uses the ranging process to determine the distance of each ONU from the OLT. After the ONUs distances and their relative single trip-delay differences are known, the OLT uses the grant assignment to ensure that upstream slots from any ONUs wouldn't collide.

In addition to measuring the logical distance between the ONU and the OLT, the ranging procedure is also used to connect new ONUs or in-service ONUs that have lost synchronization. In the case of a new ONU being connected, the OLT assigns a GEM Port-ID to it after the ranging process finishes. [2]

Dynamic Bandwidth Assignment (DBA)

The DBA is a process by which the OLT distributes the upstream transmission opportunities to the traffic-bearing entities within the ONUs, based on the dynamic indication of their activity status and their configured traffic contracts. [10]

DBA allows upstream timeslots to shrink and grow based on the distribution of the upstream traffic-entities. These have a field called T-CONTs, which are upstream timeslots, and each is identified by a particular Alloc-ID, an ONU must have at least one T-CONT, but most have several, each with its own priority or traffic class, and each corresponding to a particular upstream timeslot on the PON. Without DBA support on the OLT, upstream bandwidth is statically assigned to T-CONTs, which cannot be shared, and can be changed only through a management system. [12]

DBA enables the OLT to allocate bandwidth based on ONUs requests, or on measuring upstream traffic. As an example of a DBA process, figure 2.12 describes a scenario where bandwidth not used by Optical Network Terminations (ONTs) A and C is allocated to other ONTs that request it.

The DBA algorithm can quickly adjust the upstream bandwidth allocation according to changing traffic patterns. By allocating bandwidth according to ONUs priorities and re-allocating bandwidth from idle or low priority ONUs, a larger percentage of upstream bandwidth is used and a fair distribution of resources is guaranteed, leading to improve latency. [13]

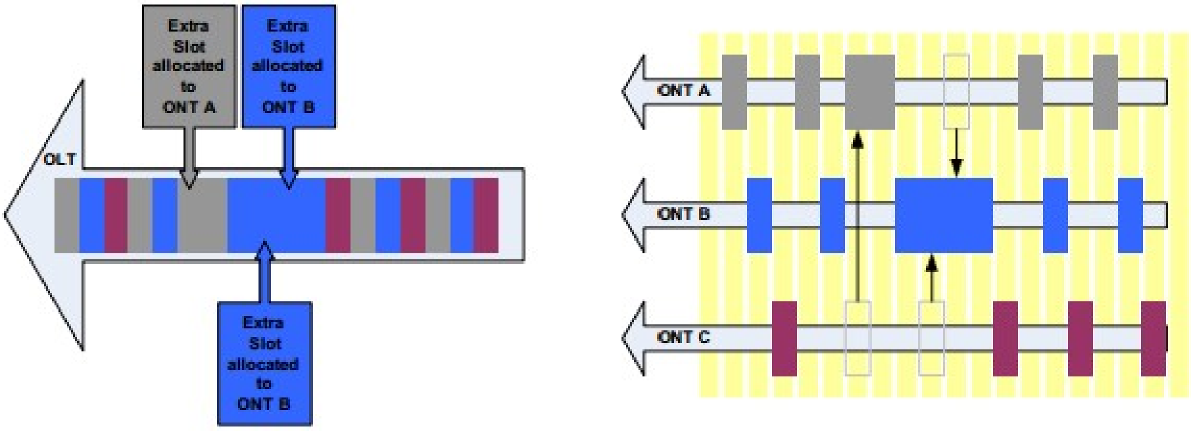

Extra Slot allocated to ONT A
Extra Slot allocated to ONT B
OLT
Extra Slot allocated to ONT B
ONT A
ONT B
ONT C

Figure 2.12: Dynamic bandwidth allocation example [13]

G-PON Protocol Stack

Figure 2.13 shows the protocol stack for the GTC layer, and is comprised of two sublayers, the GTC framing sublayer and the GTC adaptation sublayer. Furthermore, the

GTC consists of a Control and Management Plane that manages embedded Operations, Administration and Maintenance (OAM), PLOAM and OMCI features, and a User Plane that carries users traffic. In the GTC adaptation sublayer, the SDUs from GEM partitions are converted from/to conventional GEM Protocol Data Units (PDU). Also, the OMCI channel data is recognized in these partitions and interchanged to/from the OMCI client. At the framing sublayer, GEM, embedded OAM and PLOAM partitions are recognized according to their locations in a GTC frame. The embedded OAM is terminated at this layer, which handles the bandwidth granting, key switching and DBA, because information of embedded OAM is included in the GTC frame header directly. PLOAM information is processed at the PLOAM block functioning as a client of this sublayer, so that PLOAM messaging is easy to control by a specific process. [10]

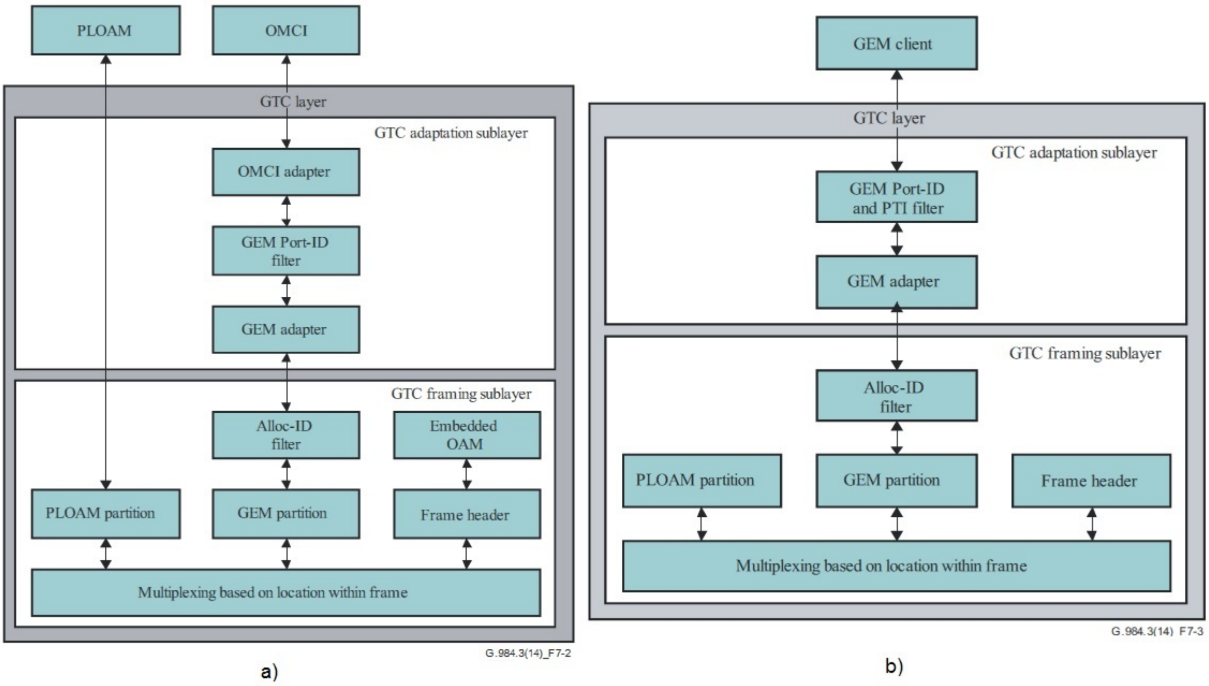


Figure 2.13: GTC Protocol Stack a) Control and Management Plane and b) User Plane [10]

2.4 40-Gigabit-capable Passive Optical Network (NG-PON2)

NG-PON2 is the response of network providers to increase the revenue through the development of new services only possible with the increase of bandwidth, especially video services that are estimated to be 90% of the actual global traffic. Furthermore, NG-PON2 is a direct evolution of PON networks, thus enabling the co-existence with legacy technologies, protecting the service providers initial investment by taking advantage of the existing ODNs.

2.4.1 Architecture

NG-PON2 is defined by the ITU-T in a series of recommendations G.989.x (x = 1, 2, 3) that define general characteristics of NG-PON2 systems and the specifications from the PMD and TC layers. On March 2013, ITU-T G.989.1 [14] was approved, addressing some requirements, and on December 2014, ITU-T G.989.2 [15] was also approved, addressing the PMD specifications.

Several technologies were suggested to support the requirements for the NG-PON2, including 40G TDM-PON, WDM-PON, Time and Wavelength Division Multiplexing PON (TWDM-PON) and Orthogonal Frequency Division Multiplexing PON (OFDM-PON). WDM-PON is excluded due the lack of backward compatibility, because they require wavelength selective ODNs. 40G TDM-PON is also excluded due to the costs of implementation for each end user and the high chromatic dispersion over long distances. OFDM-PON, due to its time-frame requirement, is a complex technology and its implementation would have high risks. As a result, according to FSAN the best solution is the TWDM-PON (TDM/WDM-PON) architecture, a system that piles up four XG-PON in only one fiber to have an aggregate capacity of 40 Gbit/s, and combines the benefits of TDM-PON and WDM-PON. TWDM-PON offers good advantages, from statistical sharing the bandwidth (so customers can flexibility get access to bandwidth ranging from some Mbits/s to peaks of 10 Gbits/s) and backward compatibility (the splitter based ODNs can be reused, reducing costs, implementation time and complexity of TWDM-PON systems). [16, 17]

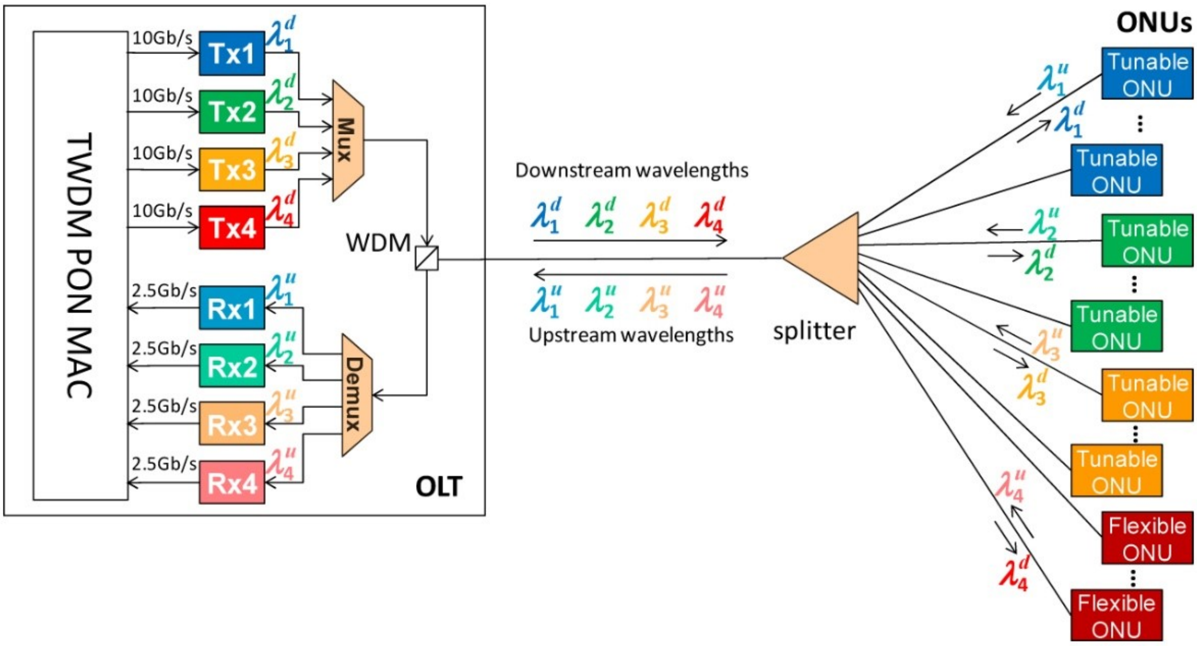


Figure 2.14: TWDM-PON architecture [18]

Figure 2.14 shows the architecture of the standard TWDM-PON system, which uses both WDM and TDMA. In such a system, multiple wavelengths co-exist in the same

ODN using WDM, and each wavelength serves multiple ONUs using TDMA. The ONU in TWDM-PON is equipped with a tunable transceiver, so it can selectively transmit or receive upstream/downstream data on a pair of upstream/downstream wavelengths.

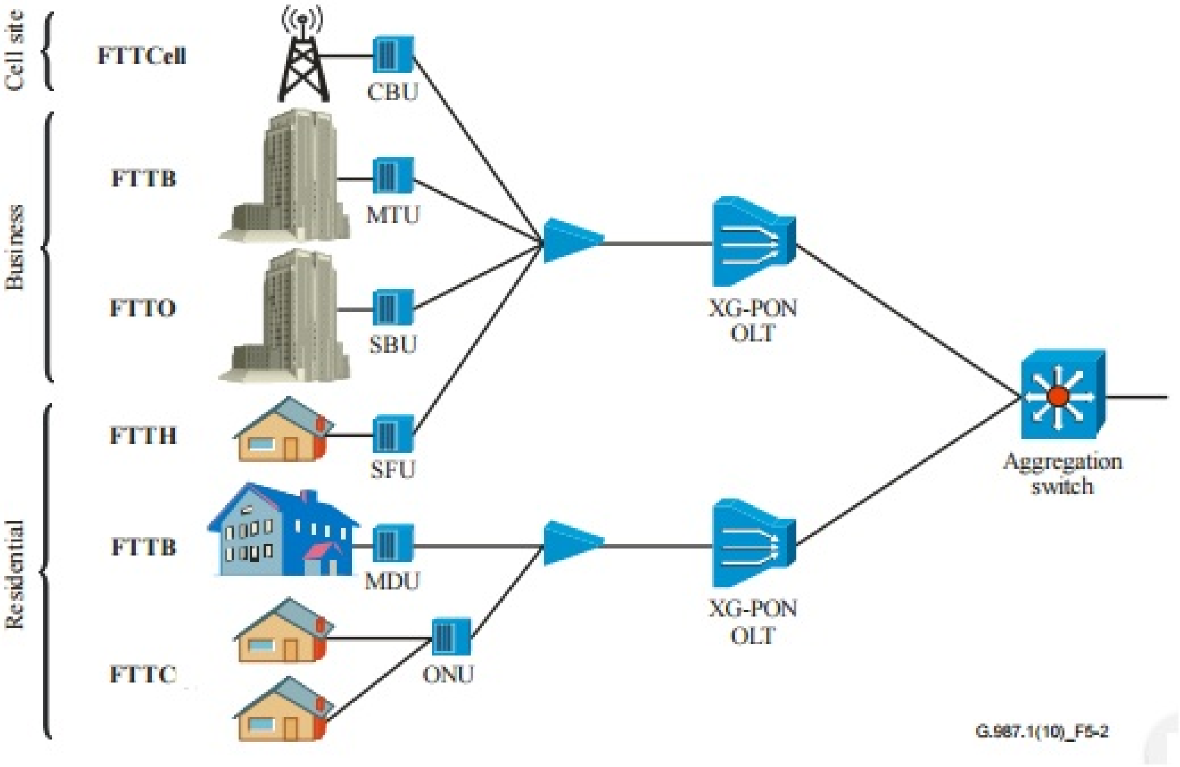


Figure 2.15: XG-PON different architectures [19]

Being a pile of at least four XG-PON, the TWDM-PON supports the same architectures as the XG-PON, described in the ITU-T G.987.1 [19] recommendation. Besides the FTTH/C/B, it's also supported Fiber-to-the-Cell (FTTCell) and Fiber-to-the-Office (FTTO). In te FTTCell scenario, the ONU is called Cell-site Backhaul Unit (CBU), and offers connectivity to the wireless base stations. The FTTO addresses business ONUs dedicated to small business customers, providing in a flexible way private line services at several rates. Figure 2.15 shown the different XG-PON architectures, and table 2.6 presents the services supported by each architecture.

	FTTH	**FTTC**	**FTTO**	**FTTCell**	**FTTB**	
					MDU - residential users	**MTU -** business users
Asymmetric broadband services	✓	✓	✗	✗	✓	✗
Symmetric broadband services	✓	✓	✓	✗	✓	✓
POTS	✓	✓	✓	✗	✓	✓
Private line services	✗	✗	✓	✗	✗	✓
xDSL backhaul	✗	✓	✗	✗	✗	✗
Symmetric TDM services	✗	✗	✗	✓	✗	✗
Symmetric/Asymmetric packet-based broadband services	✗	✗	✗	✓	✗	✗
Hotspots	✗	✗	✗	✓	✗	✗

Table 2.6: Different services supported by each XG-PON architecture [19]

2.4.2 Co-Existence Scenario

PON systems such as G-PON series ITU-T G.984 and XG-PON series ITU-T G.987 have been standardized and deployed worldwide, these are denominated legacy PON systems. Due to the major investments made in these legacy PONs (including the infrastructure), NG-PON2 systems must be able to protect this investments by ensuring a smooth migration capability for customers to NG-PON2 systems. It's expected that two or three PON generations will continue to co-exist for a relatively long time.

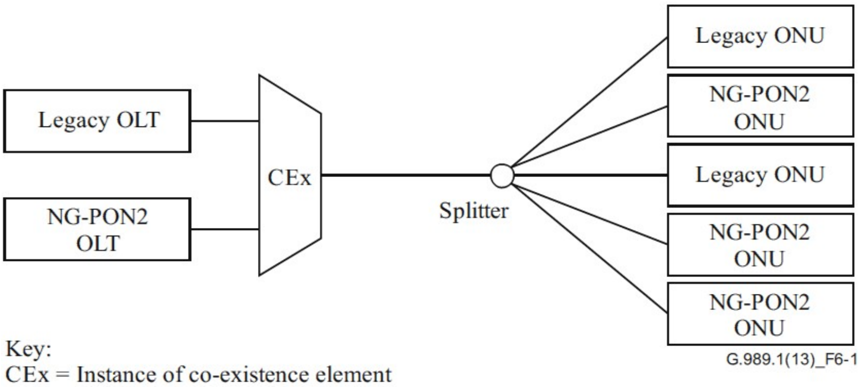


Figure 2.16: ODN co-existence scenario [14]

The co-existence scenario is enabled through the wavelength band plan, shown ahead in this chapter, and must allow co-existence over the whole, end-to-end ODN, including co-existence over the feeder fiber and optical power splitters, as seen in the figure 2.16, while also providing the optional overlay capability for Radio Frequency (RF) video signal on a separate wavelength. Co-existence facilitates a smooth migration from legacy PON to NG-PON2 systems, by enabling a flexible migration without service interruption. Also, with different transmission standards present in the PON it's required the definition of specific filter isolation between them. [14]

2.4.3 Characteristics

Bit Rate

NG-PON2 systems shall be able to support at least 40 Gbit/s aggregate capacity per feeder fiber in the downstream direction and at least 10 Gbit/s aggregate capacity in the upstream direction, the target ceiling capacity is up to 160 Gbit/s in the downstream direction and up to 80 Gbit/s in the upstream direction. Typically, any NG-PON2 ONU shall be able to support at most 10 Gbit/s service, but the actual capability per ONU on the PON will depend on engineering choices concerning split ratio adopted and the scenario considered (e.g. FTTH, FTTB, FTTCell), such service mixes are needed to enable the sharing of common infrastructure.

Future services might require different sustained and peak data rates, as well different symmetry ratios between downstream and upstream data rates, deriving different capacity requirements of the NG-PON2 system. Overall, is envisioned that NG-PON2 must be able to provide flexible levels of rate symmetry, between 2:1 and 1:1 (downstream:upstream) service rates for high levels of rate symmetry, and as low as 100:1 service rates. The NG-PON2 system will thus enable the provisioning of services that are tailored to meet different customers' needs over a common infrastructure. [14]

TWDM-PON works with 4 - 8 channel pairs, each consisting of one downstream and one upstream wavelength channel, each channel pair (downstream/upstream) should support nominal line rates of:

- 10 Gbit/s | 10 Gbit/s
- 10 Gbit/s | 2.5 Gbit/s
- 2.5 Gbit/s | 2.5 Gbit/s

Line Coding

NG-PON2 uses NRZ line coding in the downstream direction, for both 2.5 Gbit/s and 10 Gbit/s, independent of the fiber reach. In the upstream direction, for 2.5 Gbit/s, NRZ line coding is implemented for any fiber reach, while for 10 Gbit/s, NRZ is used for 20 km fiber reach, if the fiber reach extends to 40 km or more, NRZ might not be the better solution, so this case is still under study. [15]

Fibre Reach

NG-PON2 systems must support a minimum fiber reach of 40 km without reach extenders. They must also support a configurable maximum differential fiber distance of 20 km and 40 km, and they must also be backward compatible with already deployed infrastructures (CO locations, fiber cables, etc).

With reach extenders is expected a maximum reach of 60 km and even longer reaches, as long as 100 km, thus facilitating CO consolidation and other network architectures and capabilities. [14]

Split Ratio

ODNs exploiting power splitters are typically deployed with a split ratio in the range of 1:16 to 1:128. They may run over legacy power split ODNs, wavelength routing or a combination of both, thus leading to a flexible system to support cost effective deployments over the existing ODNs. NG-PON2 specific applications and network engineering choices may require higher split ratios, so the OLTs must support a split ratio of at least 1:256, and their core design shouldn't preclude supporting higher split ratios.

Support for a higher number of ONUs per ODN enables a high degree of infrastructure sharing and node consolidation if used in conjunction with longer reach, however, it leads to an increasing system complexity and power budget limitations. [14]

Wavelength Allocation

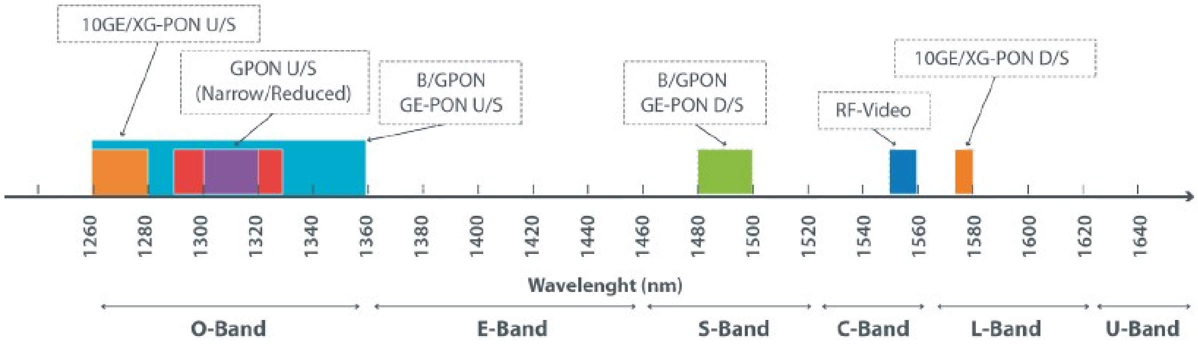


Figure 2.17: Current legacy PON wavelength plan [14]

TWDM-PON system requirements regarding the operating wavelength must always consider the possible scenario of co-existence with legacy PON wavelengths and their ODNs. When the co-existence of legacy PON systems is considered, it becomes necessary to take into account the wavelength planing which NG-PON2 technology has to co-exist with. This spectrum is shown in figure 2.17.

In the ITU-T G.989.2 [15], the wavelength plan for NG-PON2 is specified, as seen in table 2.7, enabling the co-existence through wavelength overlay with legacy PON systems.

Shared spectrum allows full coexistence with G-PON, XG-PON, RF video overlay and TWDM-PON.

Downstream	**Upstream**
1596-1603 nm	Wideband option 1524-1544 nm
	Reduced band option 1528-1540 nm
	Narrow band option 1532-1540 nm

Table 2.7: NG-PON2 wavelength bands [15]

Colourless ONUs

In order to deploy TWDM-PON systems, deployed ONUs must be colourless, meaning that they must be able to filter any downstream and upstream wavelength specified by the OLT, leading to a better flexibility and reduced OPEX in inventory management. Furthermore, the management of different ONUs types that scale in number with the number of wavelengths used on the PON is avoided, reducing the provisioning time and cost compared to coloured ONUs. [14]

In order to deploy colourless ONUs, TWDM-PON must have the ability to tune the ONU transmitter and receiver. The tunable components will be able to dynamically change ONUs transmitter operation wavelength and filter tuned wavelength, such changes must be applied by the OLT according to link conditions and QoS requirements. In TWDM-PON the wavelength band for downstream direction is specified between 1596 - 1603 nm, so the ONU-side tunable filter must operate in the L-band, for the upstream direction, wavelength band is specified between 1524 - 1544 nm in the wideband option, so the ONU-side tunable laser must operate in the C-band.

The TWDM-PON tunable transmitter and receiver deployment takes advantage of reusing mature tunable optical transport network components, helping reducing the risk of component availability. Also, wavelength tuning performance in TWDM-PON could be relaxed from that of the optical transport network, and TWDM-PON channel rates are widely used in optical transport networks, resulting in relieving critical tuning requirements, such as tuning range, tuning speed, and channel spacing. This leads to improved performances and costs reduction of producing tunable transmitters and receivers. [20]

2.4.4 NG-PON2 Transmission Convergence

NG-PON2 TC layer is based on Recommendation ITU-T G.987.3 [21]. Since NG-PON2 is an aggregation of four XG-PON fiber channels, unique modifications for NG-PON2 are required, and this modifications will be captured in Recommendation ITU-T G.989.3, which hasn't yet been approved. Some of these modifications are presented in article "NG-

PON2 Transmission Convergence Layer: A Tutorial" [22], which offers a comprehensive review of the NG-PON2 TC layer specifications.

Wavelength Management Mechanism

In TWDM-PON architecture a dynamic wavelength assignment method is needed in order to manage multiple wavelengths during the transmission process. In the Recommendation "ITU-T G.9802, Multiple-wavelength passive optical networks (MW-PONs)" [23], a wavelength assignment method is specified, where wavelength assignment has four logical functions: wavelength assignment, wavelength tuning, wavelength resource administration and wavelength channel performance supervision.

The wavelength assignment is an integral procedure of the ONU activation process, at the instant an ONU asks to join operation on the TWDM-PON, procedures such as burst profile learning, ONU-ID assignment, and ranging occur. Also, the ONU reports to the OLT the ONU wavelength and wavelength tunability, both upstream and downstream, and it's up to the OLT to make a decision and instruct the ONU to either maintain the current wavelength or tune to a different wavelength.

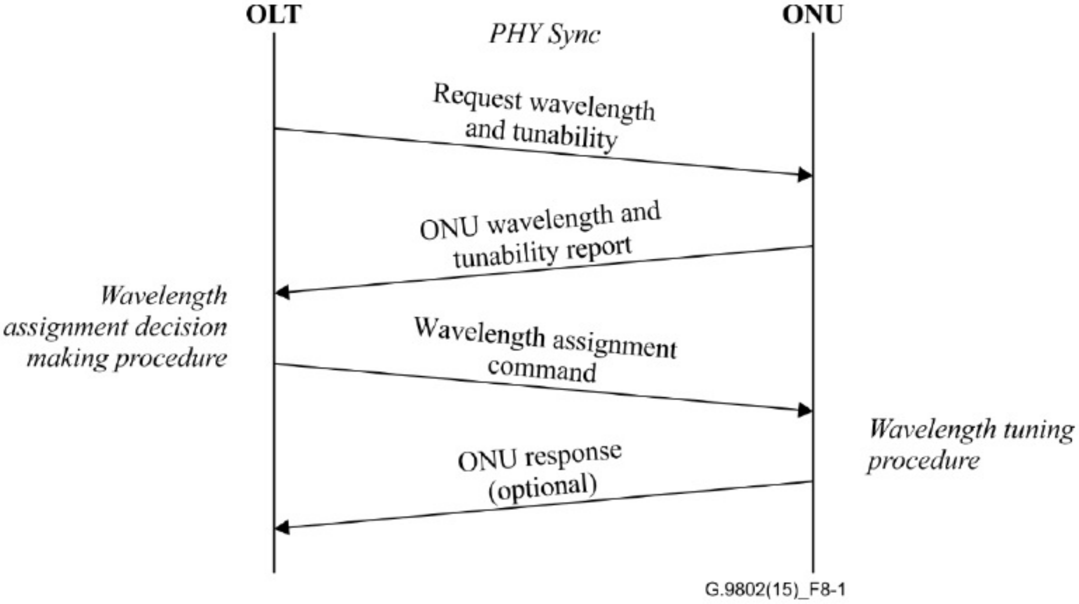

OLT
ONU
PHY Sync
Request wavelength and tunability
ONU wavelength and tunability report
Wavelength assignment decision making procedure
Wavelength assignment command
Wavelength tuning procedure
ONU response (optional)
G.9802(15)_F8-1

Figure 2.18: Wavelength assignment message exchange in ONU activation [23]

After the wavelength assignment process, the OLT can further demand an ONU to change its tuned wavelength in regular operation, this is an important process to balance PON traffic among wavelengths, moving traffic from heavily loaded wavelengths to lightly loaded or idle wavelengths, or for power saving features.

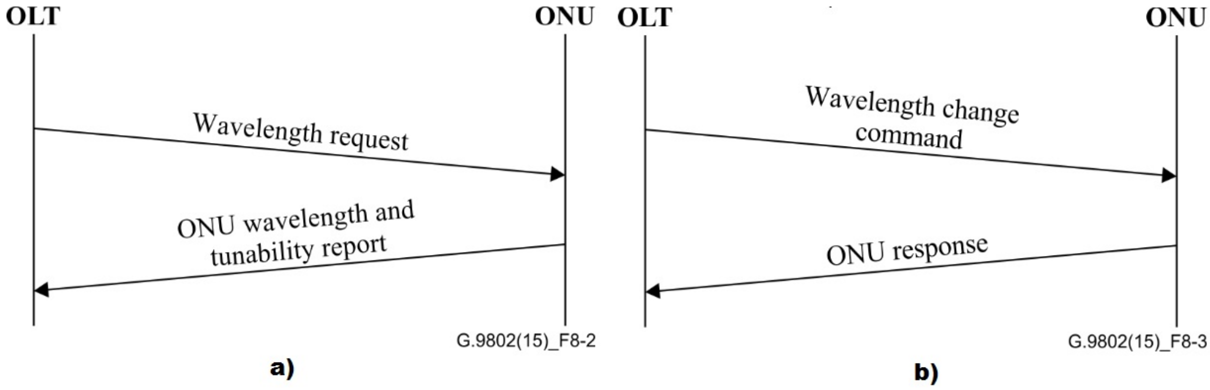


Figure 2.19: Message exchange in a) wavelength tunability request and b) wavelength tuning change [23]

The chosen method to enable the wavelength management information exchange between the OLT and the ONU is to deliver instructions over the PLOAM message channel, carrying OLT demands to the ONU and deliver ONU responses to the OLT. This method requires the introduction of new PLOAM messages in order to communicate wavelength assignment information.

In a TWDM-PON architecture, wavelengths become a key resource, identified by Logic Identifiers (Logic-IDs), so the OLT can confirm the ONU wavelength assignment in the Media Access Control (MAC) layer. This enables wavelength resource availability and wavelength assignment to be provided as a resource for administration.

Wavelength channel performance supervision supports the management and troubleshooting of the TWDM-PON. Traditional optical layer supervision described in Appendix IV of the Recommendation ITU-T G.984.2 (2008) Amendment 2 [24] should be supported via OMCI. Furthermore, in TWDM-PON, calibration information from the ONUs must be provided to the OLT for the process of wavelength tuning in ONU activation or regular operation. [23]

XG-PON Encapsulation Method (XGEM)

In XG-PON, the encapsulation method is similar to the G-PON GEM, but optimized for the XG-PON rates and TC layer. XG-PON Encapsulation Method (XGEM) is used in both downstream and upstream transmissions, supporting fragmentation, encapsulation and delineation of user data and protocol signalling SDUs. The XGEM frames are transmitted in the XG-PON Transmission Convergence (XGTC) payload sections of the downstream XGTC frames and the upstream XGTC bursts.

In figure 2.20 is shown the XGEM frame, containing a 8 bytes XGEM header and a variable size XGEM payload field containing the XGEM encapsulation of an ETH frame. In the XGEM header there is a PLI, Key Index, XGEM Port-ID, Options, Last Fragment (LF) and HEC fields. [21]

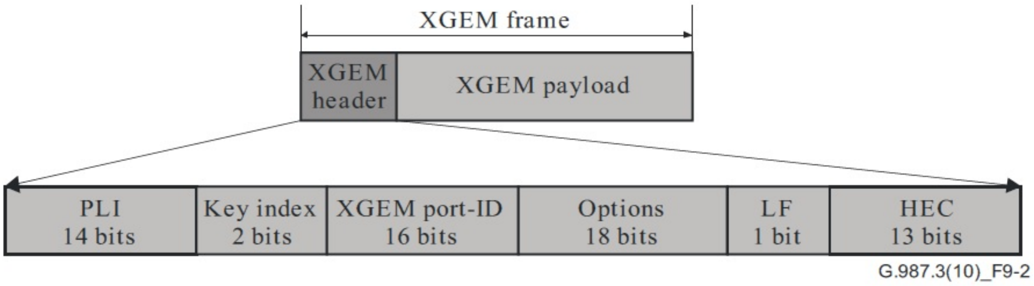


Figure 2.20: XGEM frame format [21]

Field	Functionality
PLI	Length in bytes of the SDU contained in the XGEM payload
Key Index	Indicates the data encryption key used to encrypt the XGEM payload
XGEM Port-ID	Identifies the XGEM port to which the frame belongs
Options	Unidentified
LF	Indicates whether the SDU fragment is the last fragment
HEC	Used for error detection and correction

Table 2.8: XGEM header fields

Downstream Transmission

In figure 2.21 is seen the downstream Framing Sublayer (FS) frame and the PHY adaptation frame. The downstream FS frame contains three fields, FS header, FS trailer, and FS payload containing one or more XGEM frames. The FS header has a HLend field, US BWmap and PLOAMd fields. Then the TC PHY adaptation sublayer applies the FEC in the FS frame to form the payload of the downstream PHY frame. The downstream PHY frame has a PSBd header, with a PSync, Superframe Counter (SFC) and Operation Control (OC) fields.

For the downstream transmission, the downstream PHY frames have a 125μs interval and are sent from the OLT to the ONUs. The OLT encapsulates the XGEM frames onto the FS frame payload using the XGEM Port-ID as a key to identify XGEM frames that belong to different ONUs, where the XGEM Port-IDs are given to ONUs at the activation process. Then, all the TWDM-PON OLT wavelengths are multiplexed into the fiber, and it's up to the ONUs tunable receiver to filter the correct wavelength. All the ONUs working in a given wavelength receive the same downstream PHY frames, so its up to each ONU to filter the PHY frames payload based on their XGEM Port-IDs, processing only the frames that belong to it. Multicast XGEM port-ID can be used to carry XGEM frames to more than one ONU. [21, 22]

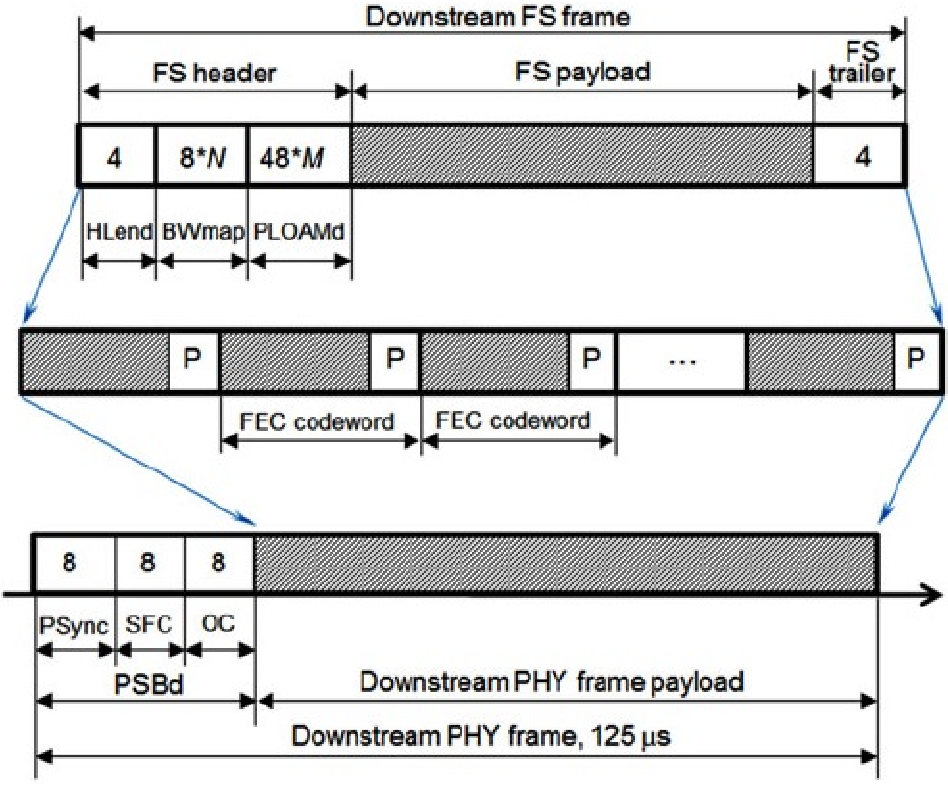


Figure 2.21: Downstream FS and PHY frames [22]

Downstream FS Frame	
Field	**Functionality**
HLend	Controls the size of the variable length partitions within the FS header
US BWmap	Used to allocate upstream transmission opportunities to the T-CONTs of the attached ONUs
PLOAMd	Contains the PLOAM message
FS trailer	Contains the BIP field, used to estimate the Bit-error Rate (BER) if FEC results aren't available
Downstream PHY Frame	
Field	**Functionality**
PSync	Used by the ONUs for downstream signal acquisition and PHY frame delineation
SFC	A counter that is incremented by one for each subsequent PHY frame
OC	Contains several fields that communicate the topological and optical power parameters of the PON system. More important are the PON Identifier (PON-ID), composed of an administrative label and the downstream Logic-ID, and also an indicator to distinguish between G.987.3 and G.989.3 framing methods

Table 2.9: Downstream FS and PHY frames headers fields [22]

Upstream Transmission

In figure 2.22 is seen the upstream FS burst and the PHY burst of the upstream PHY frames. The upstream FS burst contains the FS header, FS trailer and a series of allocations. The FS header contains a 4 octets field that has an ONU-ID, Ind, HEC fields and a PLOAMu fields. While the allocations contain one or more XGEM frames in the FS payload, and if the DBA mode is activated, the respective DBRu. Then FEC is applied in the upstream FS bursts to form the PHY burst payload. The PHY bursts have a Physical Synchronization Block upstream (PSBu) header containing a preamble (PRE) and delimiter (D) fields.

In the XG-PON upstream direction, each upstream PHY frame have a 125μs interval, containing in it one or more upstream PHY bursts. The OLT grants upstream transmission opportunities to the traffic-bearing entities within the ONUs. The ONUs traffic-bearing entities have a US BWmap field that specifies the Alloc-IDs, and the time interval for a particular bandwidth allocation. Using the TDMA protocol, the bandwidth allocations to different Alloc-IDs are multiplexed in time as specified by the OLT in the US BWmaps transmitted in the downstream direction. Within each bandwidth allocation, the ONU uses the XGEM Port-ID as a multiplexing key to identify the XGEM frames that belong to different upstream logical connections. [21, 22]

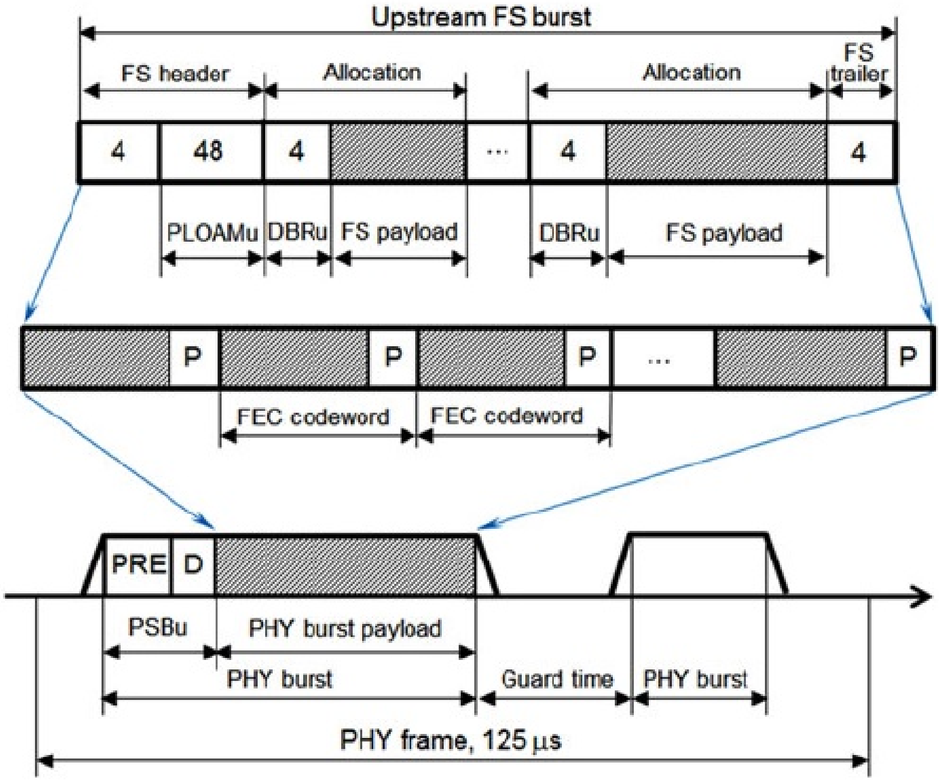


Figure 2.22: Upstream FS and PHY burst [22]

Upstream FS Frame	
Field	**Functionality**
ONU-ID	Contains the unique identification of the ONU sending the burst
Ind	Provides information regarding ONUs pending PLOAMu messages, and a Dying Gasp (DG) to indicate conditions that may prevent the ONU from proper response to upstream bandwidth allocations
HEC	Used for error detection and correction
PLOAMu	Contains the PLOAM message
FS trailer	Contains the BIP field, used to estimate the BER if FEC results aren't available
Downstream PHY Frame	
Field	**Functionality**
PRE	Used by the OLTs optical receiver to adjust to the level of the optical power
D	Used to delineate bursts

Table 2.10: Upstream FS and PHY burst headers fields [21]

Dynamic Wavelength and Bandwidth Allocation (DWBA)

The DWBA in TWDM-PON is a process by which the OLT distributes the upstream transmission bursts to the traffic-bearing entities within the ONUs, based on the dynamic indication of their activity status and their configured traffic contracts. The TWDM-PON must be able to allow wavelengths to be assigned to different ONUs dynamically, as well to allocate timeslots dynamically, like shown in section 2.3.3, performing a dynamic wavelength and bandwidth allocation.

Since there's still no recommendation on this matter, DWBA is still open for investigation, and is possible to find articles, like in [25, 26], offering solutions for this. They aim to focus on two aspects when performing DWBA, the QoS by fully using wavelengths, and power saving features by aiming to decrease the active wavelengths.

As DBA, DWBA uses T-CONTs types to determine traffic priority, each T-CONT is identified by a particular Alloc-ID. For effective DWBA, the OLT must gather information regarding the upstream bandwidth requests of each T-CONT and grant upstream transmission opportunities to them, and if necessary move ONUs to other wavelength channels if they are heavily loaded. The DWBA must guarantee that when an ONU moves to other wavelength channel, it will not try to send frames when the tuning process is occurring or while it's still sending frames.

XG-PON Protocol Stack

Figure 2.23 shows the protocol stack for the XGTC layer, and is comprised of three sublayers, the XGTC Physical Interface (PHY) adaptation sublayer, XGTC framing sublayer and the XGTC service adaptation sublayer. Furthermore, the XGTC consists of a Control and Management Plane that manages embedded OAM, PLOAM and OMCI features, and a User Plane that carries users traffic. In the XGTC service adaptation sublayer, the SDUs from XGEM partitions are converted from/to conventional XGEM PDUs. Also, the

OMCI channel data is recognized in these partitions and interchanged from/to the OMCI client. At the framing and PHY adaptation sublayers, XGEM partition, embedded OAM and PLOAM partitions are recognized according to their location in a XGTC frame. The embedded OAM is terminated at this layer since XGTC frames/bursts have well defined headers for this channel, and time-urgent control functions are sent through it, including upstream PHY burst timing and profile control, bandwidth allocation, data encryption key selection and DBA signalling. PLOAM information is processed at the PLOAM processor, functioning as a client of this sublayer, so that PLOAM messaging is easy to control by a specific process. [21]

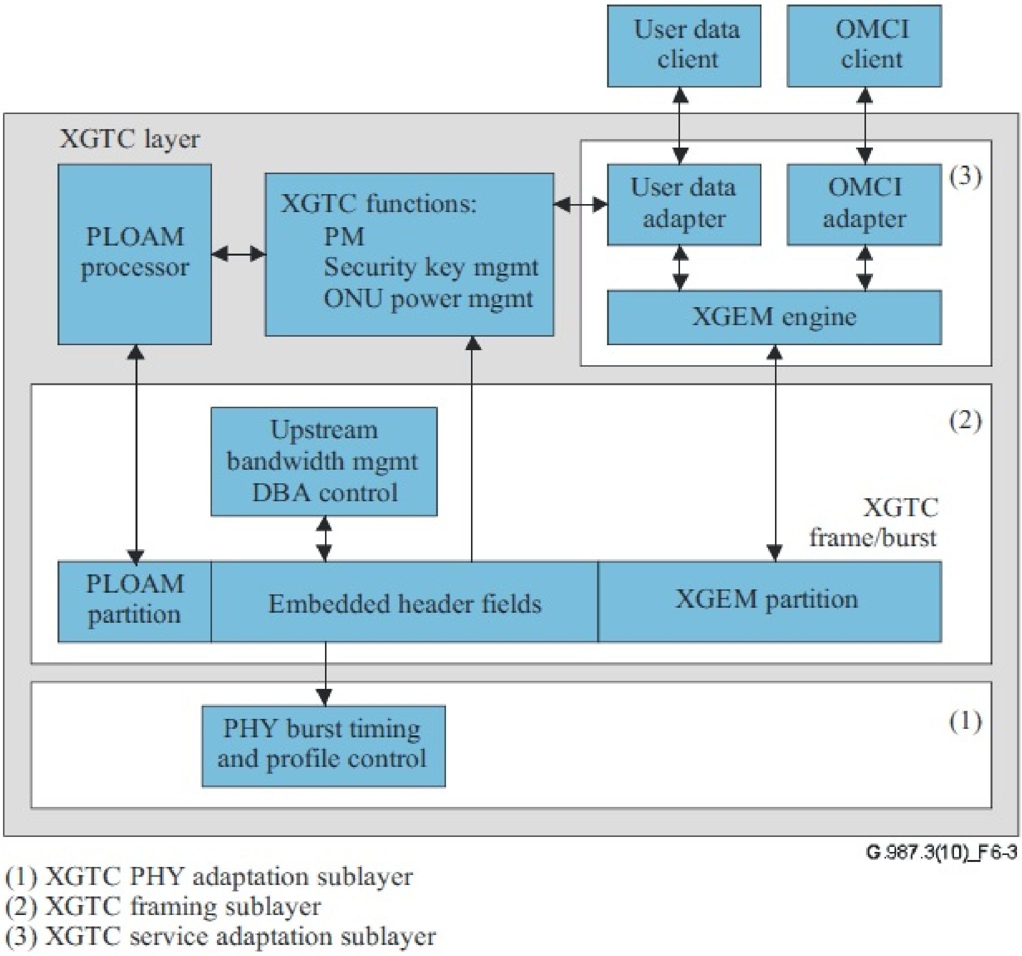


Figure 2.23: XGTC Protocol Stack [21]

2.5 G-PON/NG-PON2 Management

G-PON and NG-PON2 management mechanisms for embedded OAM and PLOAM are defined in ITU-T G.984.3 [10] for G-PON, in ITU-T G.987.3 [21] and article "NG-

PON2 Transmission Convergence Layer: A Tutorial” [22] for NG-PON2, while the ITU-T G.988 [27] defines the OMCI for both systems.

2.5.1 Embedded OAM

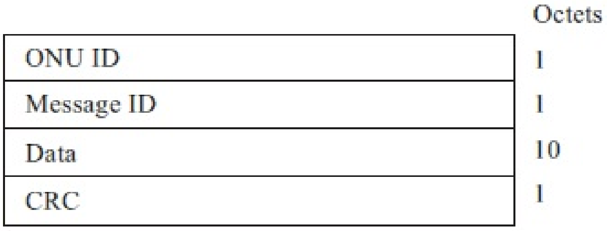
Embedded OAM is an operation and management channel between the OLT and the ONUs. It supports time urgent functions, including upstream PHY burst timing and profile control bandwidth allocation, key synchronization, DBA signalling and reporting, forced wake-up, and DG indication. [10, 21]

2.5.2 Physical layer OAM (PLOAM)

PLOAM is a message based operation and management channel between the OLT and the ONUs used for implementing PMD and GTC/XGTC layer management functionalities not done via embedded OAM. In the downstream direction, PLOAMd is used for system management and control, ONU registration and activation, ranging, alerts exchange and ONU power management. In the upstream direction, PLOAMu is an optional field, responding to management messages from the OLT in regards to ranging, ONU activation functions, and PLOAMd messages that require acknowledgement. [10, 21]

G-PON PLOAM Message

The G-PON PLOAM message structure, shown in figure 2.24, includes the ONU-ID, Message-ID, Data and Cyclic Redundancy Check (CRC).



	Octets
ONU ID	1
Message ID	1
Data	10
CRC	1

Figure 2.24: G-PON PLOAM message format [10]

Field	Functionality
ONU-ID	Used for identifying the ONU to read the message or the ONU sending it. To broadcast a message to all ONUs the value is set to 0xFF
Message-ID	Indicates the type of message and defines the semantics of the message payload
Data	Contains the payload of the PLOAM message and is specific to the Message-ID
CRC	Used to check the integrity of the message, dropping it at reception if the value is incorrect

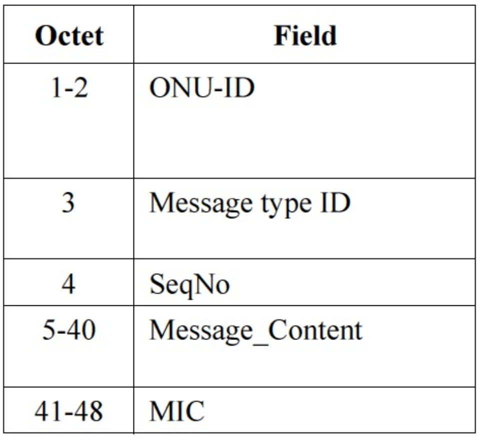
Table 2.11: G-PON PLOAM message fields [10]

To note that in the G-PON PLOAM the Port-ID for the OMCC channel must be established after the ONU activation, using the specific PLOAMd message for the process.

NG-PON2 PLOAM Message

The PLOAM message for the NG-PON2 follows the same structure as the XG-PON, supporting the XG-PON legacy functions. Additionally, for the NG-PON2 is introduced new PLOAM messages enabling system and channel profile announcement, ONU wavelength channel handover signaling, ONU wavelength channel locking, wavelength adjustment and calibration, protection configuration, optical power levelling, and rate control. [22]

The XG-PON PLOAM message structure, shown in figure 2.25, includes the ONU-ID, Message type ID, Sequence Number (SeqNo), Message_Content and Message Integrity Check (MIC).



Octet	Field
1-2	ONU-ID
3	Message type ID
4	SeqNo
5-40	Message_Content
41-48	MIC

Figure 2.25: XG-PON PLOAM message format [21]

Field	Functionality
ONU-ID	Used for identifying the ONU to read the message or the ONU sending it. To broadcast a message to all ONUs, or a sending ONU that hasn't an ONU-ID assigned, the value is set to 0x3FF
Message type ID	Indicates the type of message and defines the semantics of the message payload
SeqNo	Contains the sequence number counter that is used to ensure robustness of the PLOAm messaging channel
Message_Content	Contains the payload of the PLOAM message and is specific to a particular Message type ID
CRC	Used to verify the sender identity and to prevent a forged PLOAM message attack

Table 2.12: XG-PON PLOAM message fields [21]

In the XG-PON PLOAM the OMCC Port-ID is automatically assigned at the ONU activation procedure, taking the same value as the ONU-ID assigned.

2.5.3 ONU Management and Control Interface (OMCI)

OMCI is used for ONU management and control. A specific channel called ONU Management and Control Channel (OMCC) takes responsibility to transmit the OMCI message between the OLT and ONUs, the data units are encapsulated in the GEM/XGEM frame. OMCI message exchange usually has the highest priority in the transmission, and works in a master-slave routine. The OLT has several master entities that keep control over the multiple slave entities of the ONUs.

The OMCI is able to provide configuration management, fault management, performance management and security management.

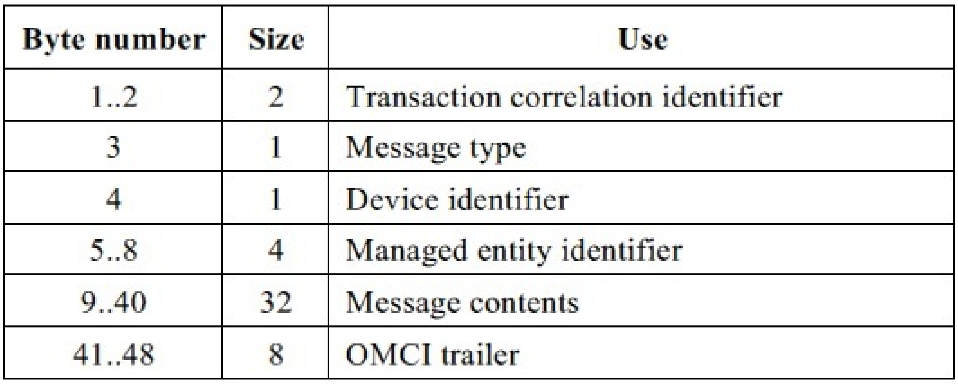
- Configuration management provides the OLT functions to control, identify, collect data from and provide data to the ONUs, being then able to configure equipments, network interfaces, GEM/XGEM ports, PHY ports, OAM flows, traffic descriptors and GEM/XGEM adaptation layer profiles.

- Fault management at the OLT is limited to only reporting failure indications presented in the network, by sending an alarm message to the network operator. Since erratic alarm messages can occur, these must be filtered before declaring them as alarms. The OLT collects alarm information from all the ONUs in real time by the OMCI fault management function, to make monitoring work easier and more efficient.

- Performance management is done by the OMCI at the request of the OLT, the OMCI then uses a subset of managed entities to retrieve the performance monitoring data of different services to the OLT.

- Security management includes the downstream data encryption function, since all downstream data is broadcasted to all ONUs, and every ONU has a reserved time slot, some users may reprogram their own ONUs to capture all the downstream data, so an Advanced Encryption Standard (AES) algorithm is used to avoid this menace. The AES guarantees that an ONU only reads the data that has the encryption key reserved to it, making it difficult to decrypt other keys. Since the upstream transmission in G-PON is point-to-point, confidentiality of the information sent is secure, and each ONU sends their own encryption key in the upstream frame, so that the OLT attach it to the downstream frames when data is sent to that ONU. To further improve security, the ONU can change their encryption key periodically without disturbing data flow.

OMCI Message



Byte number	Size	Use
1..2	2	Transaction correlation identifier
3	1	Message type
4	1	Device identifier
5..8	4	Managed entity identifier
9..40	32	Message contents
41..48	8	OMCI trailer

Figure 2.26: OMCI message format [27]

The OMCI message is encapsulated directly into the GEM/XGEM frame, or several frames if needed. The GEM/XGEM frame header contains the OMCC port-ID, and the GEM/XGEM payload has the OMCI message, and shown in figure 2.26 is the OMCI message format, containing the Transaction correlation identifier, Message type, which has 8 bytes and is subdivided into four parts, Device identifier, Managed entity identifier, Message contents and the OMCI trailer.

Field	Functionality
Transaction Correlation Identifier	Used to associate a request message with its response message
Message type	Bit 8 - reserved and set to 0 Bit 7 - states if a message needs acknowledgement Bit 6 - indicates if the message is an acknowledgement Bits 5 to 1 - indicates the message type
Device Identifier	Set to be 0x0A for a baseline OMCI message or 0x0B for an extended OMCI message
Managed Entity Identifier	Indicates which managed entity is the target of the action specified in the Message type
Message Contents	The layout of the message contents field is specific to each message type
OMCI trailer	Contains a set of specific values

Table 2.13: OMCI message fields [27]

Management Information Base (MIB)

The OMCI has a Management Information Base (MIB) used to collect and store the system managed entities information and store them at the OLT in real time. The MIB is synchronized in real-time between the OLT and ONUs, so the OLT has always the updated information regarding the managed entities. [27]

- ONU equipment management includes the ONU cardholders, power shedding, equipment protection, port mapping, remote debug, among others ONU management information;

- Access Node Interface (ANI) management is established to organize data associated with each access network interface supported by the ONU, helping the PLOAM messages more efficiently;

- Layer 2 and Layer 3 services management, including MAC bridging, Virtual Local Area Network (VLAN) tagging, multicast operations, and Internet Protocol (IP) routing and configurations;

- Services management includes IEEE 802.11, Digital Subscriber Line (xDSL), Time Division Multiplexing (TDM), voice and Multimedia over Coax Alliance (MoCA) managed entities;

- Traffic management is used to specify the description, priority and scheduling of the traffic entities;

- UNI management controls all the managed entities which are related with UNIs supported by the GEM services.

Managed Entities Attributes Cases and Actions

Managed entities attributes can be auto-instantiated by the ONU, instantiated by request of the OLT or instantiated in either way, depending on the ONU architecture or circumstances. For managed entities attributes on the ONU side is offered to the OLT the cases described in table 2.14. And on table 2.15 there are described some of the most common actions and notifications on attributes.

Cases	Instantiation Side	Description
(R)	OLT or ONU	On instantiation the ONU sets the attribute as a default value. The OLT can only read the value. During operation the ONU may update the value and notify the OLT.
(W)	OLT or ONU	On instantiation the ONU sets the attribute as a default value. The OLT can only write the value. Such attribute never triggers an OLT notification.
(R,W)	OLT or ONU	On instantiation the ONU sets the attribute as a default value. The OLT can both read and write the value. During operation the ONU may update the value and notify the OLT.
(R, Set-by-create)	OLT	On instantiation the ONU sets the attribute value via OLT create command. The OLT can only read the value.
(R, W, Set-by-create)	OLT	On instantiation the ONU sets the attribute value via OLT create command. The OLT can both read and write the value. During operation the ONU may update the value and notify the OLT.

Table 2.14: Managed entities attributes cases [27]

Actions	**Description**	**ACK**
Create	Create a managed entity instance with the set-by-create attributes values	✓
Delete	Delete a managed entity instance	✓
Set	Set one or more attributes values	✓
Get	Get one or more attributes values	✓
Alarm	Notification of an alarm or threshold crossing alert	✗
Attribute value change	Autonomous notification of an attribute value change	✗
Test	Test a specific managed entity	✓
Reboot	Reboot ONU or circuit pack	✓
Synchronize time	Synchronize performance monitoring interval time between OLT and ONU	✓
Get current data	Returns the current values of one or more attributes of a performance monitoring entity	✓

Table 2.15: Actions and notifications on attributes [27]

Chapter 3

Software Defined Networks (SDN) and Network Functions Virtualization (NFV)

3.1 Software Defined Networks (SDN)

SDN, or programmable network, makes the network control plane an independent software platform, decoupling it from the underlying forwarding/data plane of the hardware. Most networks consist of switches and routers, and they are developed by manufacturers that develop both software and hardware. Since the firmware (and other software) is developed by each manufacturer for their own equipment, the innovation of network equipments has developed quite slowly in these last years, until SDN emerged as a solution. Software has always been an important element of networking and is now the focus of innovation in networking like never before. Since SDN decouples data/forward plane and control plane, innovation can be accelerated, because software is developed independently of the hardware. The important thing is that SDN reduces network complexity, hence it's suitable for sophisticated environments, such as mobile networks. SDN is a major revolution in the internet architecture.

3.1.1 SDN Architecture

In an ideal SDN environment, the distributed control units can be centralized in one. In this approach, a controller acts as the network control, providing an abstract, centralized view of the overall network, and through the controller, network administrators can quickly and easily manage how the underlying systems (switches, routers) of the data plane will handle the traffic.

The SDN environment has also an application plane to support all the services and applications running through the network. Network intelligence is centralized in a software-based SDN controller, which enable the user to have a global view of the network, so the network appears to the applications and policy engines as a single, logical switch. The northbound Application Programmatic Interfaces (APIs) facilitate the orchestration and automation of services running atop the network platform, including routing, multicast, security, access control, bandwidth management, traffic engineering, QoS, processor and storage optimization, energy consumption, and all forms of policy management, which can be custom made to meet business objectives. The southbound APIs are responsible for defining the control communications between the controller platform and data plane services, including physical and virtual switches. The majority of network providers currently rely on the development efforts of the OpenFlow protocol for southbound communications, but the SDN architecture is flexible and can leverage other protocols. [28]

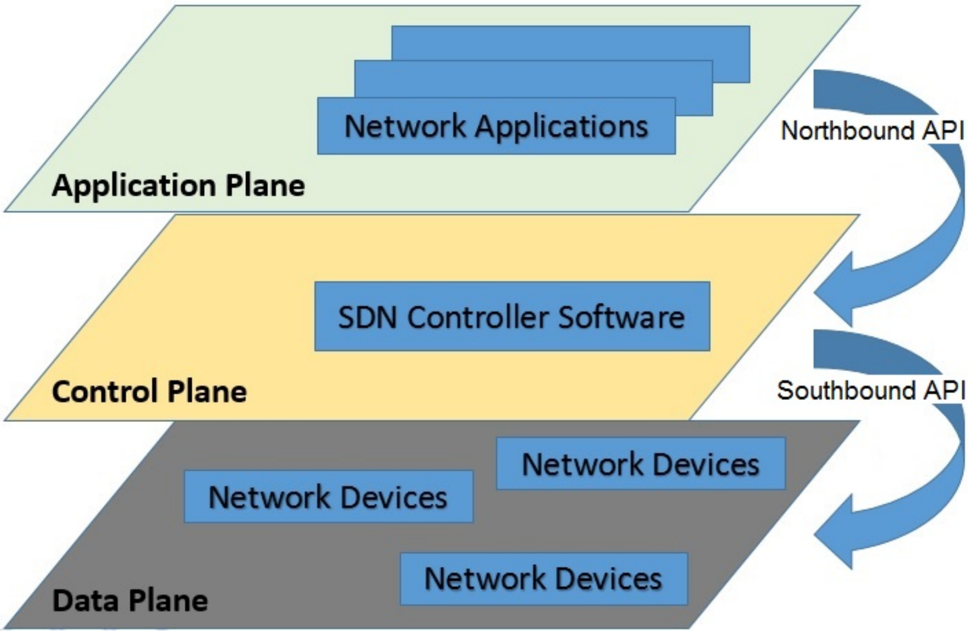


Figure 3.1: Software-Defined Network Architecture

As a result, SDN enables network administrators to shape traffic and deploy services to address the change of network requirements, without having to access each individual component of the data plane. Another great asset is the simplification of network devices, since they no longer need to process the immensity of actual protocols standards, but just accept the instructions given by the SDN controller.

Centralized SDN Control

One of the key innovation in networks introduced by separating the control and data planes is the capability to directly input SDN principles to network management, optimizing the mapping between network resources and network services and efficiently run arbitrary services on a network basis instead of operating inefficiently multiple network elements. As seen in figure 3.2, implementation of a centralized SDN controller architecture enables the abstraction of the network elements and present it to the northbound APIs with a unified view, where changes on the network demanded by the applications are communicated to the SDN controller and it's up to it to communicate those changes to all required network elements. The currently used legacy distributed control demands that changes in the network are communicated individually to each network element. With a centralized SDN control, anytime a service demands a change in the network, they are applied on-demand via software based applications.

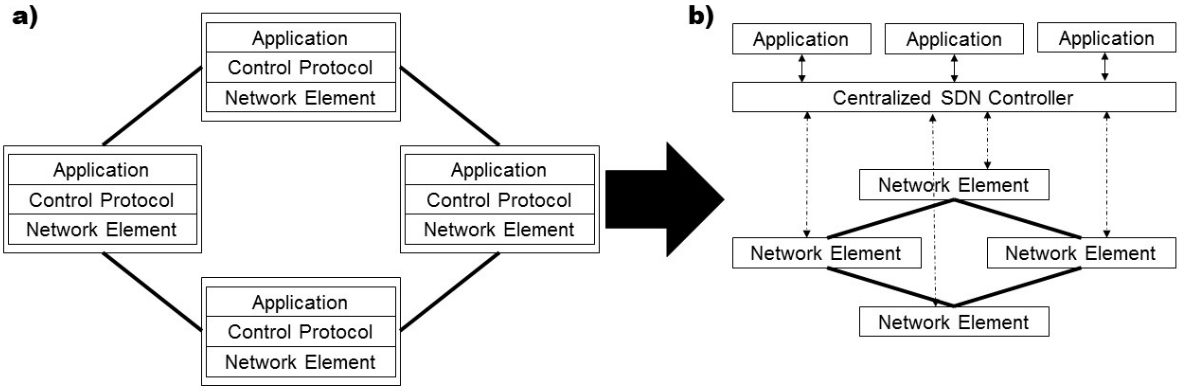


Figure 3.2: a) Legacy distributed control vs. b) Centralized SDN control

Southbound APIs

Southbound APIs are responsible for creating a more programmable network, meaning that instead of just sending commands to the devices to tell them what to do, SDN can actually reprogram the devices to function differently. The most popular protocol interface is the OpenFlow protocol, analysed ahead in chapter 3.2, and it's responsible for sending forwarding tables directly to the router, reprogramming it so that when a packet arrives at the router, it uses the table to determine its path. The problem with OpenFlow is that of fault switching. The original OpenFlow protocol, developed by the Open Networking Foundation (ONF), was projected for campus networks and is getting extensions to make it suitable for carrier networks, and since there is no consensus on SDN standards, the progress made is slow, giving rise to alternative solutions. Such solutions include: Network Configuration Protocol (NetConf) [29] which uses an Extensible Markup Language (XML) to communicate with the switches and routers to install and make configuration changes; and Lisp [30], also promoted by the ONF, that supports flow mapping. [31]

Another important aspect of southbound APIs is the ability to control different technologies, not just multiple vendors equipment. Networks are composed of different devices that manage specific segments of the network. Taking into consideration the TWDM-PON architecture, that have specific wavelengths for different types of access media, such as FTTH and wireless services, which associated equipments to backhaul must also be managed by the SDN controller.

Northbound APIs

Northbound APIs are of critical importance in the SDN environment, since the value of SDN is tied to the innovative applications it can potentially support and enable. Northbound APIs must support a wide variety of application types, not only transport functions, but also load balancers, firewalls or other Software Defined Security (SDSec) services, or orchestration applications across cloud resources for example.

SDN main application is the capability to compute paths across the network, or end-to-end provisioning. Path computation is highly dependable of the southbound APIs, they must be able to work with hardware with disparate protocols, and also be able to manage systems that don't have embedded controllers. Another important application is data flow provisioning, where the SDN controller must know in real time what resources are available in the network, not just information written in a database that can be outdated, thus leading to an efficient control of the network. Northbound APIs can then be programmed to build the commands needed to tell the SDN controller what the endpoints, bandwidth demands, and restrictions of the needed services are, and the SDN controller can automatically provision them. Another area where SDN can improve network utilization is the optimization of the network, in an autonomous and automatically way, performing connection tests whenever possible, searching for protection and restoration paths and communicating them to the underlying hardware whenever needed. [32, 33]

Overall, northbound APIs open interface means that users can build their own applications and don't need to wait on equipment vendors to deliver features that they need now, making possibilities practically limitless. More than that, there are applications that can cross multiple domains that could never have been addressed in the past.

Eastbound-Westbound APIs

SDN architecture offers the possibility of multiple SDN controllers being deployed over a single network, where each SDN controller manages different devices or sections of the network. The purpose of the eastbound-westbound APIs is to manage interactions between the various SDN controllers.

3.1.2 SDN Classes

Depending on the value proposition, three distinct classes of SDN have emerged, with unique missions and different applications. Flow Services SDN addresses the security and services that become possible with flow-level programmability. Virtualization SDN provides virtual network connectivity for efficiency and agility. And Infrastructure SDN exposes the programmability of to software applications. [34]

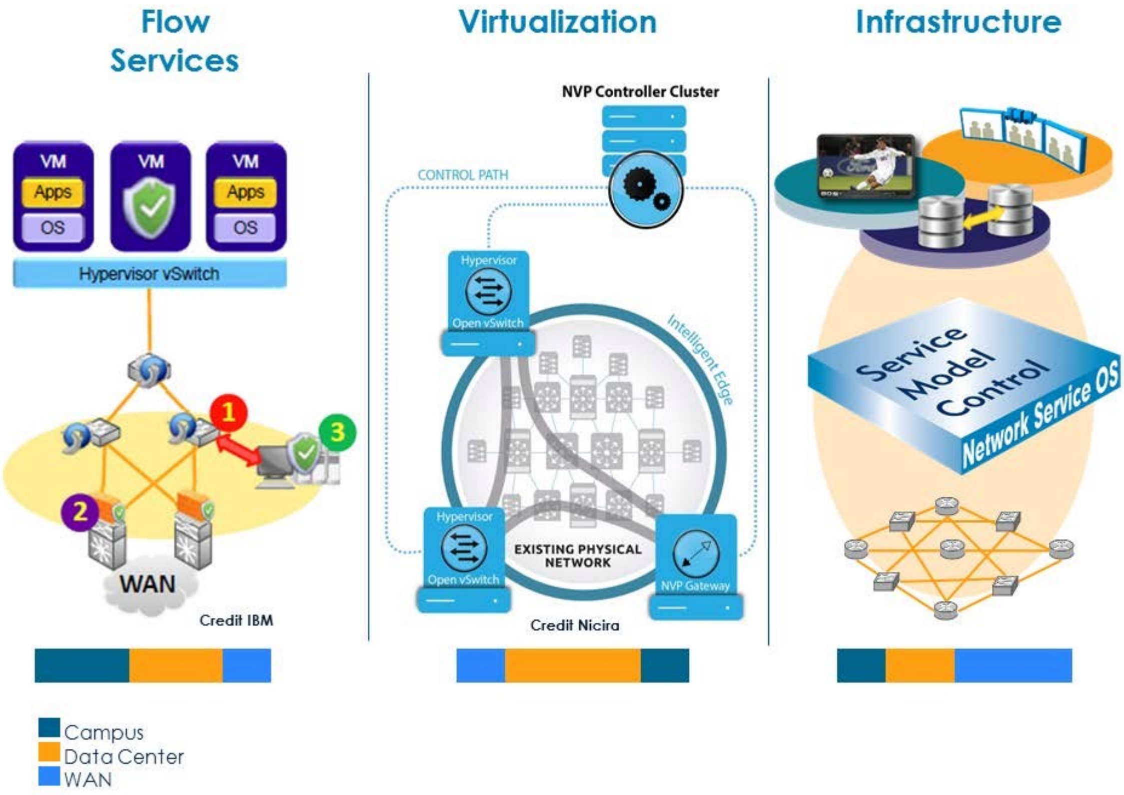


Figure 3.3: SDN classes [34]

Flow Services SDN

It's used for software applications to communicate with individual data flows or an aggregation of flows, this means an easy communication with applications requiring flow-level visibility and streamlined security. SDN makes it possible to program a functionality that specializes in packet processing platforms for control of the flows in a simple way, instead of resorting to specialized instrumentation platforms like sFlow and NetFlow.

This innovation is perfect for small enterprise business and service providers in small networks, like an university campus, data center perimeters, Wide Area Networks (WANs) and so on. In cases where it's dealt mostly with an aggregate of flows and it's needed to see and control microflows, Flow Services SDN is less significant. [34]

Virtualization SDN

Aims to make the orchestration of changes easier to manage for network equipments (firewalls, Virtual Private Network (VPN) aggregators, switches and routers), and it will revolutionize the way we manage servers in data centers. One principle of virtualization is to decouple the virtual and physical aspects of the network. With a virtualization layer there is no need to deal with the physical aspects of the network, since those aspects are emulated by Virtual Machines (VMs).

Nowadays we have virtualized services like cloud services and virtual networks, which can save up space in personal computers but require robust data centers to manage all the data needed. [34]

Infrastructure SDN

It draws value from providing visibility and programmability to physical network resources, it's the key to manage resources that get constrained because of expenses, geographic or space limitations, or there are dynamic changes, like failures, that aren't under the control of the network operator.

It's of interest to service providers to increase operational efficiencies to fight the declining average revenue per customer and increase the profit margin through new and better services. It's now considered a matter of survival for services providers to thrive on the telecommunications market, where the competition is harsh, and the possibility of services differentiation can make a big difference for enterprises to proliferate.

The main benefit of infrastructure SDN is the capability to provide information, access and control of network resources that were impossible to access in real time until now. Data centers now have the benefit of being able to reserve large chunks of bandwidth for a limited period of time. [34]

3.1.3 SDN Benefits

SDN efficiently separates the control logic from packet forwarding, allowing users to utilize very complex forwarding rules. SDN make it possible for the network to be a competitive differentiator, plus a way to reduce CAPEX and OPEX. With the ability to address the high-bandwidth and dynamic nature of modern applications, it makes it possible to adapt the network to business needs, and significantly reduce management complexity.

In the end this provides the following benefits: [35, 36]

- Reduce CAPEX - reduces the need to purchase networking hardware and supporting pay-as-you-grow network models to eliminate wasteful overprovisioning.

- Reduce OPEX - enabling the control of the network, through network elements that are programmable, makes it easier to design, deploy, manage and scale networks. The ability to automate provisioning and orchestration of data flow not only reduces overall management time, but also reduces the chance of human error in this process. Network services can be packaged for application owners, freeing up the networking team.

- Agility and Flexibility - SDN creates flexibility in how the network can be used and operated. Enterprises can create new applications, services and infrastructures to quickly meet their needs using standard development tools, and can selectively increase computing resources according to those needs.

- Easy Upgrade with a centralized computing system, upgrades and patches can easily be applied through the entire network. Furthermore, integration of new devices to the network becomes simpler, since communication of changes is applied in the whole network instead of being done individually in each equipment.

- Improved Uptime by eliminating manual intervention, SDN enables the reduction of configuration and deployment errors that can impact the network.

- Better management - service providers can use a single viewpoint and toolset to manage virtual networking, computing and storage resources. It can simplify very complex protocol processing.

- Planning - better visibility into network, computing, and storage resources means that enterprises can plan strategies more effectively for their customers.

- Innovation enabling enterprises to create new types of services and applications that can create new revenue streams and return more value from the network.

- Sustainability - some of the existing devices (computers, power supplies, routers, etc.) have poor efficiency, which is environmentally and economically undesirable. Through allocating the resources more reasonably and utilizing more efficient equipment, SDN can reduce energy consumption, compared with traditional architectures.

This benefits show why companies need to move to SDN and virtualized networks. Especially with the shifting to mobile networks, the need for cost reduction and better network management demands a new perspective on the way we see the network.

3.2 OpenFlow

3.2.1 Architecture

OpenFlow is the first protocol designed specifically for SDN, and is already standardised by the ONF [37]. Its concept was first proposed in the paper "OpenFlow: enabling innovation in campus networks" [38], and it aims to provide a high performance traffic control across multiple vendors network devices and is responsible for the communication between the control and forwarding layers of the SDN infrastructure. This innovation allows the direct access to and manipulation of the underlying network hardware (e.g., switches, routers) via a programmable interface, making possible to verify new network protocols or topologies without modifying the underlying network equipment.

In actual networks, switches are responsible for handling high-level routing and packet forwarding according to an inherent *flow table*. The control plane is decoupled from the physical network and placed in a centralized controller that uses OpenFlow to communicate with all other components in the network, thus enabling the handling of the network as a whole rather than individual devices, allowing a more effective use of network resources. The control of the *flows table* configuration is specified according to the "OpenFlow Switch Specification v. 1.3.0" [39].

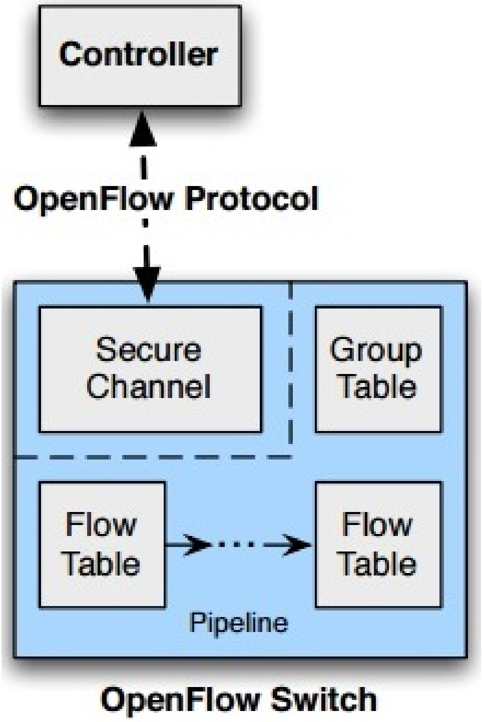


Figure 3.4: OpenFlow architecture [39]

For implementing this architecture two requirements must be met. First, a common logical architecture in all the switches, routers and every equipment in the managed SDN controller, so it sees a uniform logical switch function. Second, a secure protocol is needed between the SDN controller and the network devices. As seen in figure 3.4, a SDN controller communicates with an OpenFlow switch via OpenFlow channel, and the communication must be encrypted using a Transport Layer Security (TLS), or it can operate directly over a Transport Control Protocol (TCP). Each switch connects to other OpenFlow switches and/or to the end-user devices that are the sources and destinations of packet flows. [39]

3.2.2 OpenFlow Controller

The OpenFlow controller is the entity responsible for managing how to handle traffic without valid flow entries, and at the same time maintaining all the network protocols and policies, distributing appropriate instructions to the network devices. The OpenFlow switches establish a communication channel with the controller via IP address using a specified port, then initiates a standard TLS or TCP connection to the controller. The traffic between the controller and switch doesn't go through the OpenFlow pipeline, so it must identify when incoming traffic is from the controller before matching it with the *flow tables*. Each OpenFlow switch may establish communication with a single or multiple controllers. [39]

3.2.3 OpenFlow Flow Tables, Match Fields and Actions

The packet flow through the switches is managed according a *flow table*, implemented via hardware or firmware, and OpenFlow will program the forwarding plane of the switches.

OpenFlow protocol uses the concept of flows to identify network traffic based on predefined rules programmed by the SDN controller, so it allows to define how the traffic should flow through network devices based on parameters such as usage patterns, applications and cloud resources, thus enabling the network to respond in real time to changes in it. This isn't possible on current IP-based routing because all flows between two endpoints follow the same path through the network and if the network changes due to some kind of problem, these networks consumes lots of resources to find the problem and change the path.

The OpenFlow *flow entries* are represented by six field parameters, described in table 3.1.

Parameters	**Description**
Match fields	Used to match packets based on their ingress port and packet headers
Priority	Used to define different priorities for different *flow entries*
Counters	Used for statistics purpose
Instructions	Instructions applied in the matching packets that arrive on the switch, resulting in changes to the packet, action set and/or pipelining processing. Required instructions are the **Write-Actions *action(s)*** and the **Goto-Table *next-table-id***
Timeouts	Used to define how many time a *flow entry* can be active before expiring, and how many seconds of traffic absence is necessary before a rule expire
Cookie	Used to filter flows statistics, modifications and deletions, mainly for controller use

Table 3.1: Openflow field parameters [39]

OpenFlow has optional actions to support VLAN tagging functionality, being able to push, pop, or set up VLAN tag headers in the outgoing traffic of an OpenFlow flow. In access networks, where traffic differentiation is done through the Q-in-Q technology, where packets contain a double-tagged VLAN field to differentiate Home Area Network (HAN) through the Customer VLAN (C-VLAN) and services with the Service V-LAN (S-VLAN), and the possibility to match and manipulate this tags is important to transport multiple customer segments or VLANs across Layer 2 infrastructures. Also optional is the matching of tunnel IDs, that is defined by logical port implementation, and if received by a physical port has a value of zero.

OpenFlow tables can perform simple actions based on rules following a match structure, the rule is defined by picking a packet parameter and match it with a pre-determined value(s). OpenFlow supports the matching of L2-L4 headers from packets against *flow entries*, efficiently abstracting hardware switching into a *flow table*. Required match fields, plus VLAN tagging and tunnelling, are presented in table 3.2.

Match fields	**Description**
OXM_OF_IN_PORT	Ingress port. This may be a physical or logical port.
OXM_OF_ETH_DST	Ethernet destination MAC address.
OXM_OF_ETH_SRC	Ethernet source MAC address.
OXM_OF_VLAN_VID	VLAN-ID from 802.1Q header.
OXM_OF_VLAN_PCP	VLAN priority code point (PCP) from 802.1Q header.
OXM_OF_ETH_TYPE	Ethernet type of the OpenFlow packet payload, after VLAN tags
OXM_OF_IP_PROTO	IPv4 or IPv6 protocol number.
OXM_OF_IPV4_SRC	IPv4 source address. Can use subnet mask or arbitrary bitmask.
OXM_OF_IPV4_DST	IPv4 destination address. Can use subnet mask or arbitrary bitmask.
OXM_OF_IPV6_SRC	IPv6 source address. Can use subnet mask or arbitrary bitmask.
OXM_OF_IPV6_DST	IPv6 destination address. Can use subnet mask or arbitrary bitmask.
OXM_OF_TCP_SRC	TCP source port.
OXM_OF_TCP_DST	TCP destination port.
OXM_OF_UDP_SRC	UDP source port.
OXM_OF_UDP_DST	UDP destination port.
OXM_OF_TUNNEL_ID	Metadata associated with a logical port.

Table 3.2: Required match fields [39]

By redefining the ”address” as a digital value instead of a physical interface, new methods of support and optimization of services arise.

OpenFlow switches required actions are shown in table 3.3.

Actions	**Description**
Output	Forwards a packet to a specified OpenFlow port
Drop	Packets whose action sets have no output action and no group action must be dropped
Group	Process the packet through the specified group
Push VLAN header	Push a new VLAN header onto the packet. The Ethertype is used as the Ethertype for the tag. Only Ethertype 0x8100 and 0x88a8 should be used.
Pop VLAN header	Pop the outer-most VLAN header from the packet
Set VLAN ID	Modify the values of outer-most VLAN header field in the packet

Table 3.3: OpenFlow required actions [39]

Packets processed by the OpenFlow switch are forward to ports, which can be physical ports, logical ports, and reserved ports. Physical ports are switch defined ports that correspond to a hardware port of the switch, logical ports are switch defined ports that don't correspond directly to a hardware port of the switch, while reserved ports specify generic forwarding actions, such as sending the packet through all ports the switch can use for forwarding (port **ALL**), send to the controller (port **CONTROLLER**), or through any port (port **ANY**).

Furthermore, OpenFlow switches processes all packets by the OpenFlow pipeline, every one of them contains one or more *flow tables*, which in turn contain one or more *flow entries*. *Flow tables* are numbered in the order packets traverse through them, in each *flow table* the packets are matched against the *flow entries* and the instruction set is executed, then the packets are directed to another *flow table* using the **Goto-Table** instruction, where the same process is repeated again, as seen in figure 3.5. Packets can only be directed to a *flow table* numbered above the current one, and the last table on the pipeline stage doesn't include the **Goto-Table** instruction. When a packet isn't directed to a *flow table*, the pipeline processing stops and the packet is processed with the associated action set and usually forwarded. [39]

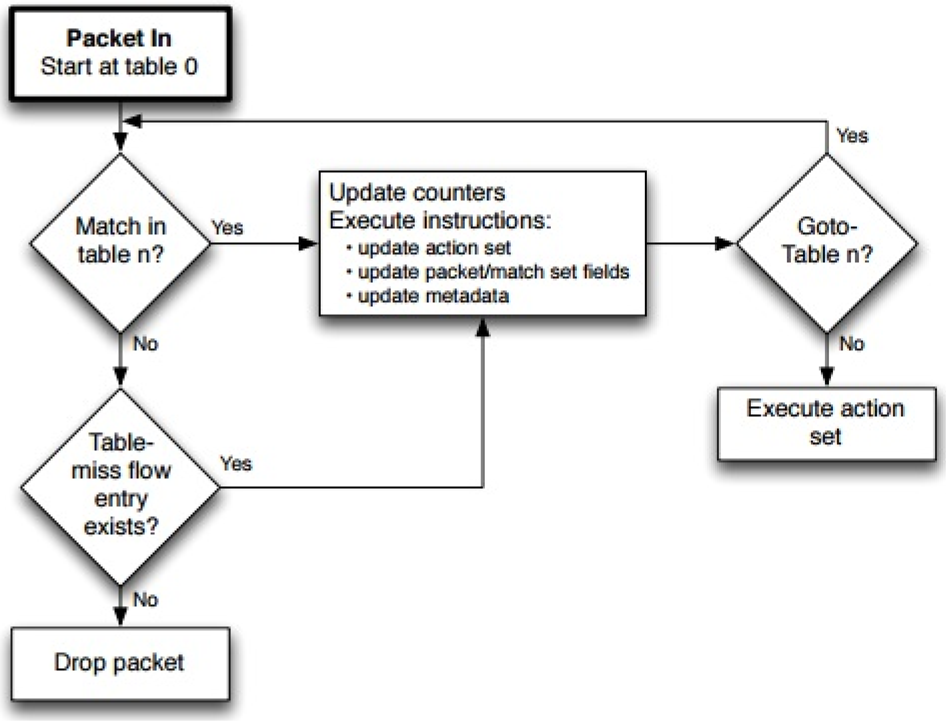


Figure 3.5: Flowchart detailing OpenFlow matching and instruction execution in a *flow table* [39]

3.2.4 Protocol Messages

The OpenFlow protocol supports three types of messages: [39]

- Controller-to-Switch: These messages are initiated by the controller and, in some cases, require a response from the switch. This class of messages enables the controller to manage the logical state of the switch, including its configuration and details of flow and group table entries. Also included in this class is the **Packet-out** message, used when a switch sends a packet to the controller for it to decide if the packet is dropped or directed to an output port of the switch.

- Asynchronous: These types of messages are sent without solicitation from the controller from a switch. Includes the **Packet-in** message, which may be used by the switch to send a packet to the controller when there is no *flow table* match; the **Flow-Removed**, that is used when a *flow entry* is removed from a *flow table* by a flow delete request; the **Port-status**, sent when port configuration states changes; and the **Error**, to notify the controller of problems encountered.

- Symmetric: These messages are sent without solicitation from either the controller or the switch. They are simple yet helpful. **Hello** messages are typically sent back and forth between the controller and the switch when the connection is first established. **Echo** request and reply messages can be used by either the switch or controller to measure the latency or bandwidth of a controller-switch connection or just verify that the equipment is operating. The **Experimenter** messages are used to stage features to be built into future versions of OpenFlow.

3.2.5 Open vSwitch (OvS)

An Open vSwitch (OvS) is an open source implementation of a virtual switch compatible with OpenFlow that functions within an hypervisor and provides connectivity between virtual and physical Network Interface Controllers (NICs). This leads to the OvS to forward traffic between different VMs and/or from a physical network to a VM. It implements standard ETH switching, and additionally enables the export of interfaces for manipulating the forwarding state and managing configuration state at runtime.

Figure 3.6 illustrates the architecture of the Open vSwitch, constituted by two main components, the userspace ”slow path” and a kernel ”fast path”. The majority of the functionality is implemented within the slow path, where packets that flow through this path are matched against OpenFlow rules which can be added by an OpenFlow controller (external interface) that communicates with the userspace, then these rules are installed in the kernel for similar future flows to go through the fast path. Furthermore, the userspace connects with a OvS Database (OvSDB) management protocol (external interface) [40] that is responsible for performing the management and configuration of OvS instances, where each instance can support multiple logical data paths, referred to as ”bridges”. The fast path processes packets with rules already defined by the userspace and doesn’t invoke any other parts of the OvS, and if packets arriving don't match any flow defined in the kernel *flow table* they are forced onto the slow path to be processed. [41, 42]

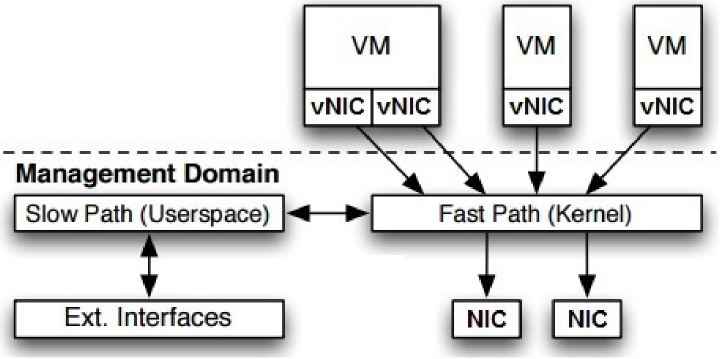


Figure 3.6: OvS Architecture [41]

3.2.6 OpenFlow Benefits

OpenFlow based SDN will provide a more flexible and programmable architecture than conventional networking architectures, in this way, new features and applications can be added to networks more easily, furthermore, there is a possibility to improve various networking areas, even complex networking tasks and features. This way, network providers can be competitive through service differentiation instead of cost differentiation. For multi-vendors environments, SDN centralized control software can control any OpenFlow enabled network equipment from any vendor, including switches, routers, and virtual switches. SDN orchestration and management tools quickly deploy, configure, and update devices across the entire network, instead of relying on the management from individual vendors. Furthermore, OpenFlow enables a flexible network automation and management, by deploying tools to automatize management tasks done manually until now, reducing OPEX and decreasing the chances of network instability due to human error.

SDN makes it possible to increase security and network reliability for service providers, since the controllers have a complete visibility and control over the network, they can ensure that access control, traffic engineering, QoS, security, and other policies are enforced consistently across all the infrastructure. Policies can be applied at various network layers, including application, session, and network devices layers, in an abstracted and automated way. Also, with OvS, operators can support multi-tenancy while maintaining traffic isolation, security, and elastic resource management when customers share the same infrastructure.

3.3 Network Functions Virtualization (NFV)

The European Telecommunication Standards Institute Industry Specification Group (ETSI ISG) has been the responsible for developing requirements and architectures for

NFV within telecommunications networks. The group was launched in January 2013 when it brought together seven leading Telecom network operators. Its purpose focuses on setting requirements and architecture specifications for hardware and software infrastructures needed to make sure virtualized functions are maintained, and also manage guidelines for developing NFs. [43]

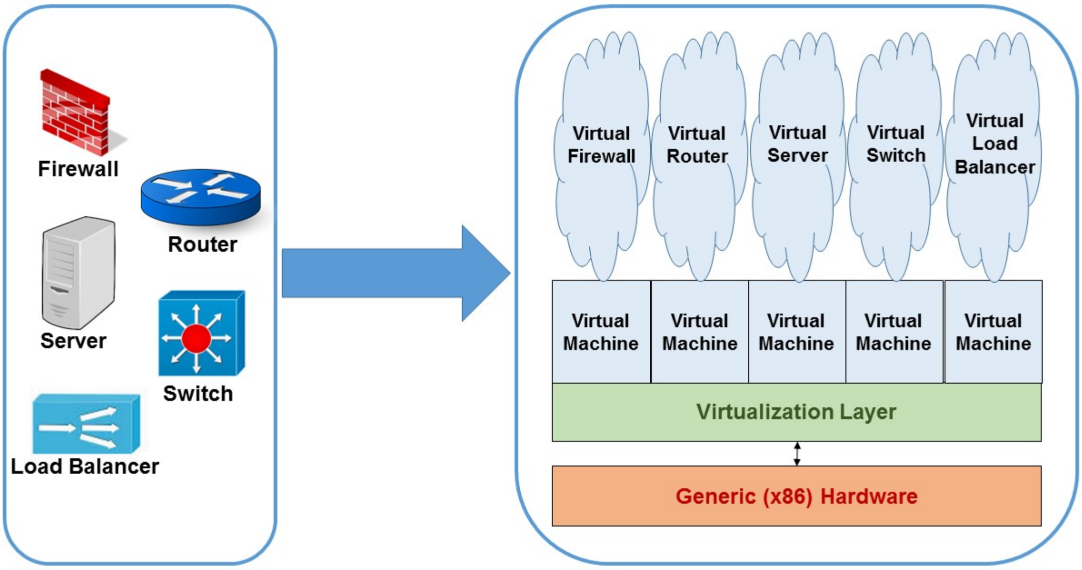


Figure 3.7: NFV goals

Network Virtualization goal is to take all of the network features, services and configurations necessary and provide it to the applications virtual network - VLANs, Virtual Routing and Forwarding (VRF), firewall rules, load balancer pools, Virtual IPs, IP address management, routing, isolation, multi-tenancy, etc. - take all of these features from the physical plane and move it to the virtualized software layer. By reproducing these Logical switches, Logical routers, Logical Load Balancers, Logical Firewalls, and more, assembled in any arbitrary topology, thereby presenting the virtual compute a complete virtual network topology.

Due to the virtual environment, networks become more simple, flexible and scalable, leading to OPEX and CAPEX reduction. Also, with each specific feature of the network running through an independent virtual server, testing new applications becomes easier and with lower risks, and so time to market of applications is reduced. The decoupling of software from hardware elements enables the evolution of both independently from each other. The network becomes more flexible since at a given time software and hardware can perform different functions and network software functions can become automated. [44]

The virtualization of a network equipment enables that functions belonging to it can be transferred to a data center owned by the service provider, replacing the equipment with a standard and simpler piece of hardware, relieving costs and enabling changes to functions to be done instantaneously, without the need of a technician to move to the equipment location.

3.3.1 NFV Architecture

The NFV architecture according to the ETSI ISG envisages the implementation of NFs as software-only entities that run over the NFV Infrastructure (NFVI). The NFV architecture, identified in figure 3.8, has three main working domains, Virtual Network Functions (VNFs), NFVI and NFV Management and Orchestration (NFV MANO). The NFVI includes the physical resources of the network, and the mechanisms in which these are virtualized; the VNFs are the software implementations of network functions, which are capable of running over the NFVI; the NFV MANO focuses on all virtualization-specific management tasks necessary by the NFV functional blocks.

The ETSI ISG for NFV also identifies a series of functional blocks within the NFV architecture, also seen in figure 3.8 and explained in the following section.

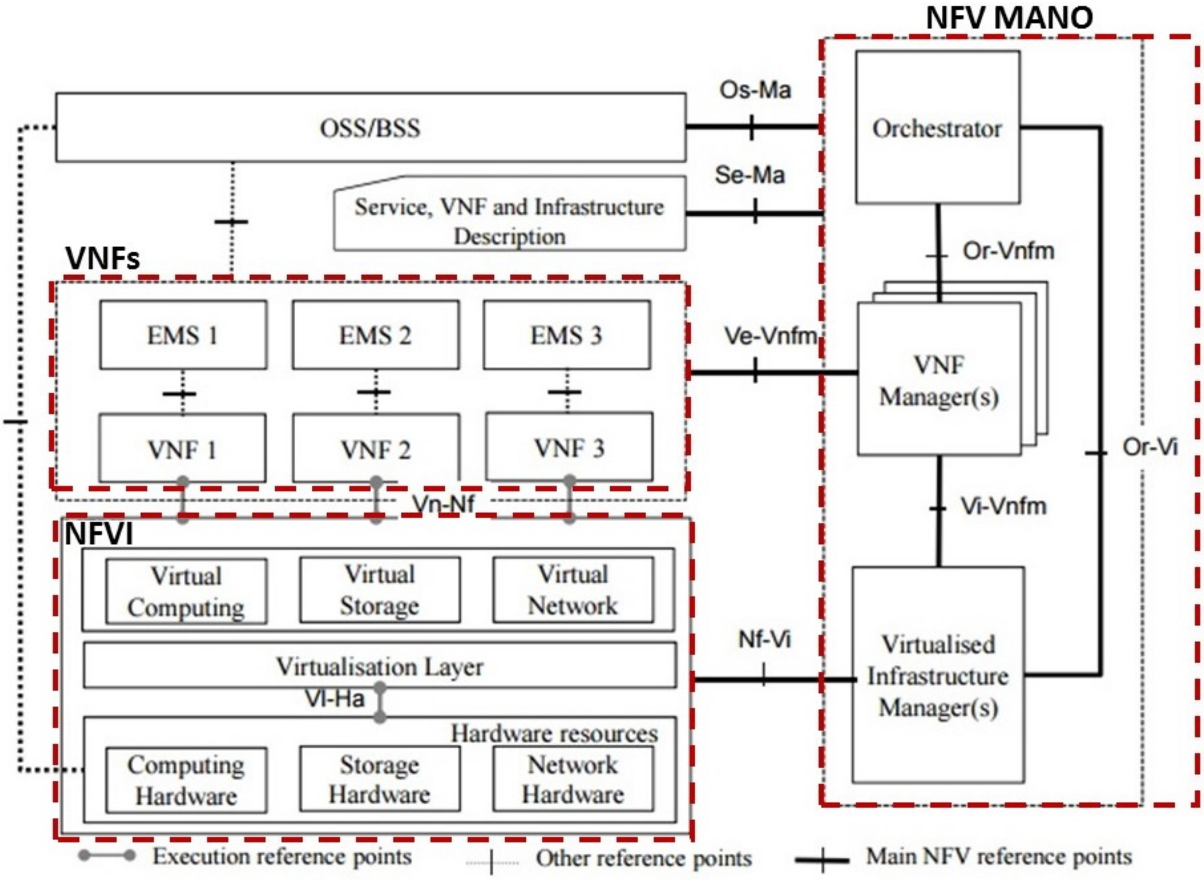


Figure 3.8: NFV architecture [44]

NFV Infrastructure (NFVI)

The NFVI includes the totality of all hardware and software components in which the VNFs are deployed, managed and executed. The VNFs see the NFVI as a single entity that provides them the desired virtualization resources. The physical hardware resources include the computing, storage, and network elements that provide processing, storage and connectivity to VNFs through the virtualization layer. The virtualization layer abstracts and logically partitions physical resources, enabling the software that implements the VNF to use the underlying Virtualized Infrastructure (VI) and provides virtualized resources to the VNF, that latter can be executed. The virtualization layer can be deployed with the use of OvS based hypervisors, or with software running on top of a non-virtualized server by means of an operating system. With the virtualization of the network domain, computing and storage resources may be represented in terms of one or more VMs, while network hardware is abstracted in order to compute virtualized network paths that provide connectivity between VMs of a VNF and/or between different VNF instances. A possible way of network virtualization is centralizing the control plane of the transport network, separating it from the forward plane, isolating the transport medium (e.g. in optical wavelengths). [44]

Virtualized Network Function (VNF)

The VNF is a virtualization of a network function in a legacy non-virtualized network, such as the Residential Gateways (RGs), firewalls, etc. The VNF can be deployed over multiple VMs, where each VM hosts a single component of the VNF, or it can be deployed in a single VM. The service providers uses both VNFs and non-virtualized NFs and offers services to the users, which expect the services to perform as if there is no virtualized elements in the network. The number, type and ordering of VNFs that make it up are determined by the services functional and behavioural specification. Therefore, the behaviour of the service is dependent on that of the constituent VNFs. The Element Management System (EMS) is responsible for performing the management functionalities of one or several VNFs. [44]

NFV Management and Orchestration (NFV MANO)

NFV MANO provides the functionality required for the provisioning of VNFs, and the related operations, such as the configuration of the VNFs and the infrastructure these functions run on. It's responsible for VNF lifecycle management (e.g. instantiation, update, query, scaling, termination), and the orchestration and management of physical and/or software resources of the VI. It also includes a data-set that provides information regarding the NFV deployment template, VNF Forwarding Graph, service related information, and NFV infrastructure information models. In addiction, it offers the functionalities that are used to control and manage the interaction of a VNF with hardware resources and their virtualization. As a resource manager it's in charge of inventory the software and hardware resources dedicated to the NFV infrastructure, allocation of virtualization enablers (e.g.

VMs onto hypervisors) and management of infrastructure resource and allocation (e.g. increase resources to VMs, resource reclamation, and energy efficiency). And offers operations for visibility into and management of the NFV infrastructure, root cause analysis of performance issues from the NFV perspective, collection of infrastructure fault informations and collection of information for capacity planning, monitoring and optimization. [44]

The VI manager relies on cloud computing resource managers, such as the OpenStack, to control the assignment of compute, storage and network VMs resources. The VNF manager(s), given the multiplicity of VNFs that can be deployed, must be provided by VNF manufacturers, and able to establish communication with the software chosen as the VI manager.

3.4 OpenStack

3.4.1 OpenStack Architecture

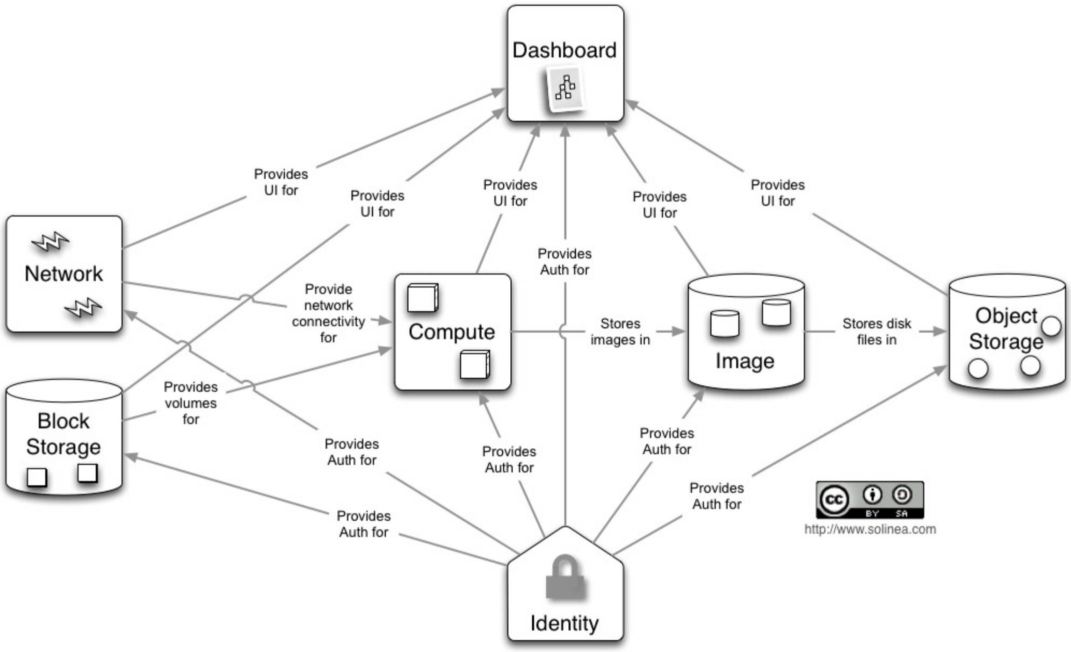


Figure 3.9: OpenStack conceptual architecture [45]

OpenStack is a public and private cloud manager tool, used to control large pools of compute, storage, and networking resources throughout a data center, and therefore recognized as the best solution for the VI manager in the NFV architecture. It's designed to provide flexibility in cloud designing, without proprietary hardware or software requirements and the ability to integrate with legacy systems and third party technologies. The OpenStack development began in 2010 as a joint collaboration from NASA and Rackspace

Hosting as a project to enable massively scalable infrastructures. It's now under active development of the OpenStack Foundation, an independent and non-profit corporation founded in September 2012 to promote OpenStack software and its community. [46]

Figure 3.9 shows a stylized and simplified view of the OpenStack framework from the operators side, assuming that the most common configuration is used. It's divided into multiple components, each providing a specific and well defined service, and accessible through a proper API.

3.4.2 OpenStack Services

OpenStack is comprised of multiple projects, where each focus on a particular cloud computing service. They are developed semi-independently of each other, avoiding dependencies between them as much as possible. This multi project architecture allows administrators to deploy cloud scenarios using only the services they need for its implementation. In this section is provided an overview of the seven most common core cloud services and the interaction between them. [45, 47]

- Network (Neutron): provides a pluggable, scalable and API-driven system for managing networks and IP addresses. Due to its plugin architecture it can accommodate different network equipments and software, providing network models for different applications or users group. It's responsible to setup virtual switches, launch Domain Name System (DNS) servers, configure routing tables, and other related tasks that enable the deployment of VMs on isolated virtual networks. It's also capable of interacting with external network devices in order to setup the physical network on a data center and to handle VMs traffic.

- Compute (Nova): responsible for providing the management of VMs life cycle within the OpenStack cloud, giving a platform to compute resources, networking, authorization, and scalability as the cloud needs. By itself it doesn't have any virtualization capabilities, but is needed to interact with supported hypervisors.

- Block Storage (Cinder): responsible for deploying virtual block storage devices, called Cinder volumes, that are then available to VMs as storage blocks and managed by Nova. Multiple instances of Cinder volumes can be deployed, each with different characteristics, for example as a high performance database storage or as a backup storage of a volume stored in Swift.

- Image (Glance): is a lookup and retrieval system for VMs images, providing an efficient and easy way to boot VMs. It can be configured to use the Swift or a local file system as a storage backend.

- Object Storage (Swift): implements a reliable, distributed, massively scalable storage model alternative to the Cinder, providing an API to store and retrieve individual objects (e.g. generic files, disk images, etc).

- Dashboard (Horizon): is the terminal in which users and operators interact with the OpenStack framework. It's web based and can be used to implement most of the functions provided by the OpenStack components, interacting with them through specific APIs that offer a management interface to Horizon.
- Identity (Keystone): provides authentication and authorization policy services for all OpenStack components of the cloud. Authentication verifies that a request actually comes from who it says it does and authorization verifies whether the authenticated user has access to the requested service.

An important aspect of the OpenStack services is that they depend on other technologies to do their work, as seen above Nova depends on hypervisors, Neutron on virtual switches, Glance on object storage, and except for the Horizon all they depend on a database. In a way to make OpenStack framework generic and flexible, all the components have the notion of plugins. An example is the Neutron, where there is a plugin for OvS, Linux Bridge, or Modular Layer 2 (ML2). This plugin aspect is of critical importance since it increases the extensibility of the OpenStack framework, enabling it to adapt to the requirements of different infrastructures. [45]

3.4.3 OpenStack Neutron

On the current dissertation the most important OpenStack project is the Neutron. It allows vNICs from Nova managed VMs to be connected to each other by providing the fundamental resources for their connection, described in table 3.4.

Resources	**Description**
Network	Consists on the abstraction of a typical L2 network segment
Subnet	A block of IPv4 or IPv6 addresses and associated to a network. Other network configurations associated with the subnet include default gateways, Domain Name Servers (DNS), and other L3 attributes.
Port	A port analogous to an attachment port on a L2 network, such as the NIC of a network switch. When created by the Neutron, an available fixed IP and a MAC address is automatically assigned from an available subnet.

Table 3.4: Resources associated with the OpenStack Neutron [45]

These resources after implemented can be used by other OpenStack components, like the Nova, to attach virtual devices to ports on these networks. Being plugin based, the Neutron relies on vendors plugins to provide the backend implementation needed to execute the network topology. The plugins can then use a variety of technologies to implement requests, like OpenFlow, VLANs, tunnel set-ups, and more.

OpenStack Neutron and ML2 Plugin

One requirement of this dissertation is the integration of the SDN controller into the OpenStack Neutron. For this it's necessary to use the ML2 plugin, which was designed to allow the Neutron to simultaneously utilize a variety of L2 network management technologies. As seen in figure 3.10, the plugin provides two types of functions, the first one is the Type Driver, responsible to manage network types related information (e.g. VLAN IDs when using VLANs or tunnel IDs when using GRE, etc), and the second one is the vendor specific Mechanism Driver, responsible for handling the interaction with external network controllers and communicate to them changes on the network. For the desired purpose, the Mechanism Driver has the corresponding SDN controller driver, enabling to create/update/delete/read operations on networks, subnets and ports resources. On the SDN controller runs an OpenStack Neutron agent establishing a communication channel with the ML2 plugin via a Representational State Transfer (REST) API using northbound communication.

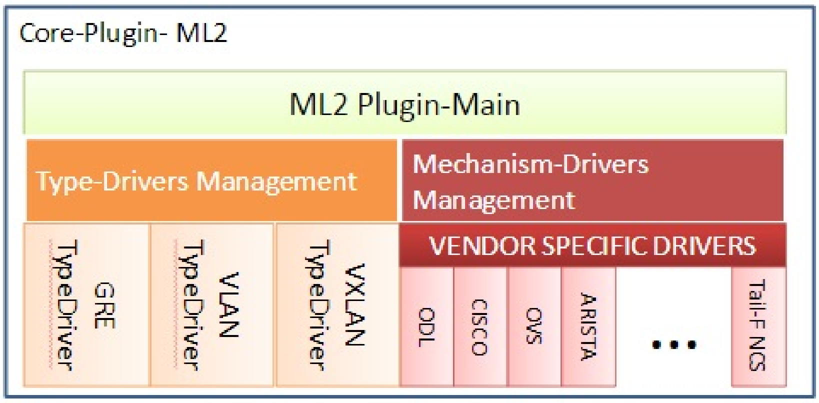


Figure 3.10: ML2 plugin architecture [48]

Changes on the network are communicated to the OpenStack Horizon, translated into the corresponding networking API and sent to the Neutron, that on receiving this requests passes them to the configured plugin. The Neutron/plugin database is then updated and the plugin invokes the corresponding REST API to the SDN controller. The SDN controller on receiving the requests perform the necessary changes in the network equipments through OpenFlow or OvSDB protocols. [48]

3.4.4 OpenStack Nodes

OpenStack deployment relies on a three-node architecture scenario, these being the Controller node, Network node and at least one Compute node, as can be observed in figure 3.11. [45, 49]

- Controller Node provides the centralized and unified management system for OpenStack deployments. At least is responsible to manage authentication and messaging to all other systems through the message queue. It is possible to have more than one controller node deployed, so if a node fails another can take over the required tasks. The services managed by the controller are: databases, message queue services, conductor services, authentication and authorization for identity management, image management services, scheduling services, user dashboard and API endpoints.

- Network Node is responsible for dedicated networking operations, running the Dynamic Host Configuration Protocol (DHCP) agent, L3 agent and the plugin agents used. From this node it's provided to VMs the connectivity to external networks, and the VMs connectivity to the network node is enabled by a L2 agent, like the OvS agent, that sets up GRE tunnels [50] interconnecting the VMs from multiple compute nodes that are part of the same network. This node provides a virtual router that provides L3 forwarding and Network Address Translation (NAT) for tenants' VMs.

- Compute Node hosts the VMs managed by the Nova component, providing the resources necessary to run VMs, such as processing, memory, network and storage. This is the most deployed node in the network because of the number of VMs that need to be deployed in a multi-tenant environment.

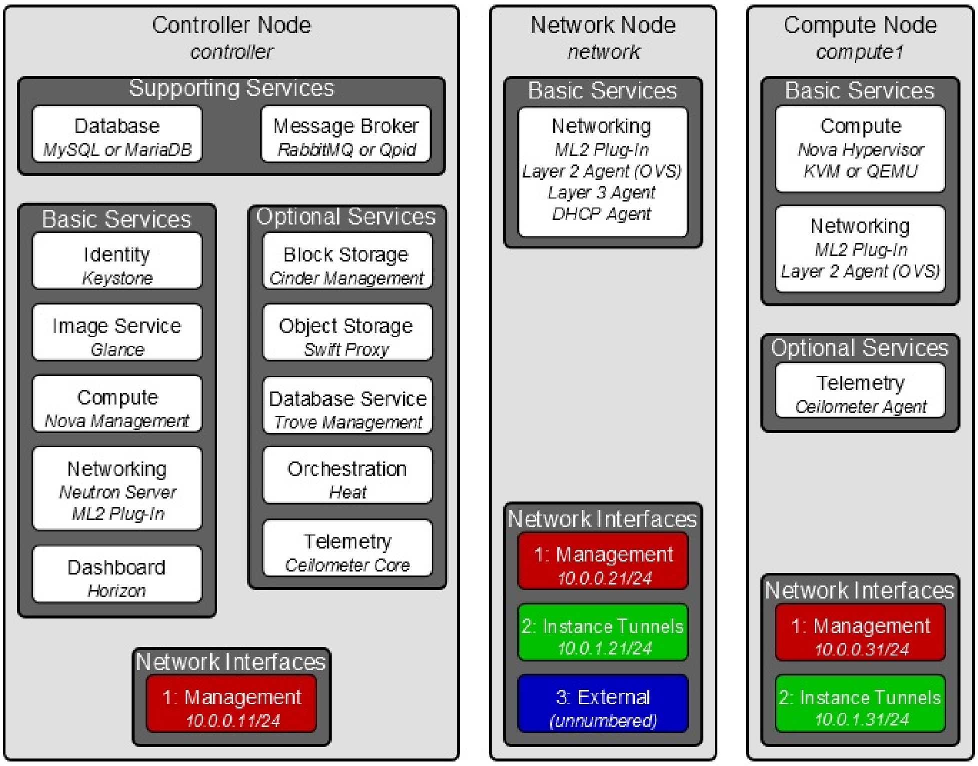


Figure 3.11: OpenStack Neutron three node deployment [51]

3.4.5 OpenStack Networks

In a typical OpenStack three node deployment there are at least three types physical networks, being that the most typical deployments rely on four, that can be seen in figure 3.12 and explained below. [52]

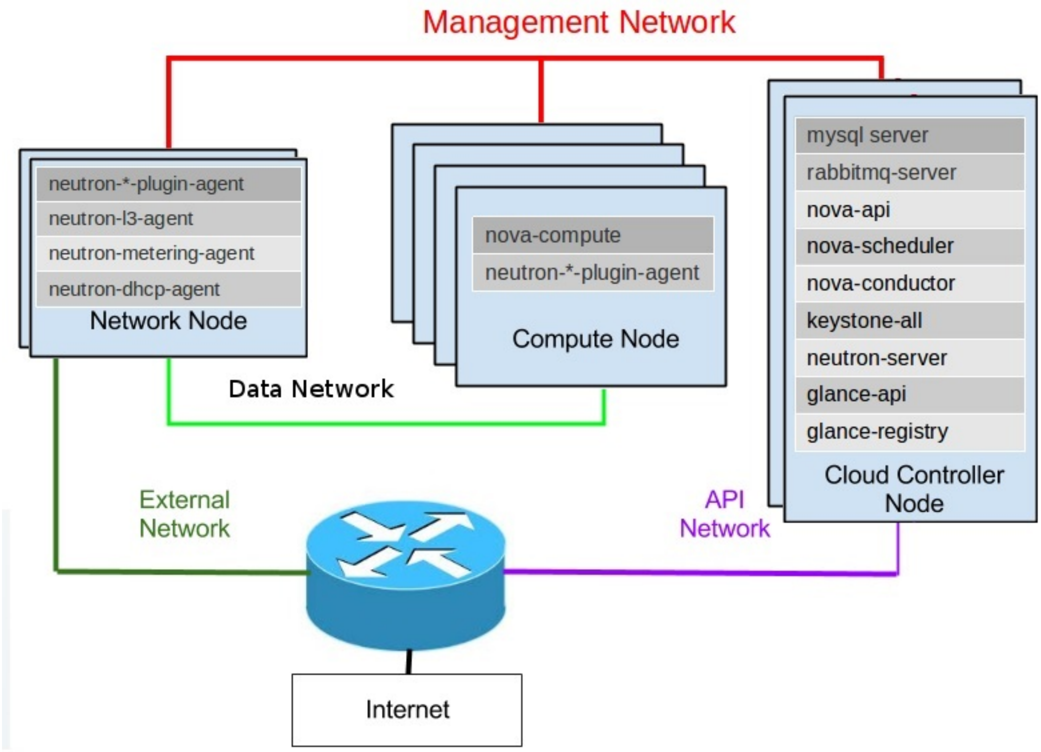


Figure 3.12: OpenStack networks in a three node deployment [51]

- Management Network is used for the internal communication between OpenStack components. This network is deployed to provide administration and monitoring of the system without being disrupted by tenants' traffic, and as such the IP addresses associated with it must be only reachable within the data center.

- Data Network is used for VMs data communication within the cloud deployment. The IP addressing requirements of this network depend on the active Neutron plugin, that establishes tunnels between VMs. Usually each tenant has a dedicated data network for its nodes, prevent the interference of traffic from other networks.

- External Network is a network directly connected to the network node that provides connectivity to the Internet. This network connects with the virtual router running on the network node, allowing access the external network to tenants' VMs by having them also connect with the virtual router. This network relies on the virtual router to act as a NAT server so VMs can communicate with it by using the pool of public

IP addresses, names floating IPs, that are assigned to VMs via NAT rules and thus can be identified from the external network.

- API Network exposes all OpenStack APIs to be accessed from anyone on the Internet. However, it should only be accessed for administration purposes, and as such it must have improved security measures.

Chapter 4

Software Defined Access Networks

4.1 Introduction

The study made in chapter 2 shows that NG-PON2 will use the TWDM-PON technology, and the defined wavelength plan has different wavelengths than the G-PON, meaning that both technologies can co-exist in the same ODN. Service providers typically use one fiber infrastructure to serve shared PONs and another to serve other business customers who desire dedicated services. The addition of multiple wavelengths allows service providers to serve business customers that require dedicated services using the same physical fiber infrastructure as residential customers. This way the same fiber infrastructure can reserve different wavelengths for FTTcell, FTTH, FTTB for business and residential customers, among other distribution topologies, delivering the respective service within the same fiber infrastructure, thus reducing fiber construction costs. Figure 4.1 shows the expected evolution of the access infrastructures, integrating both G-PON and TWDM-PON OLTs, the different types of access routers that can communicate with the OLT in a single CO, and the different types of ONUs depending on the distribution topology and services provided. In resume, it is seen that the aggregation and access network function sets have a lot of equipments that bundled together can difficult management and escalate both CAPEX and OPEX for service providers. Due to this complexity, SDN and NFV considerations can play a important role on the simplification of the access network, especially in the COs infrastructure by reducing the quantity of routers and switches and by moving control functions from hardware elements into generic servers.

In chapter 3 is shown that SDN and NFV architectures have no restrictions on what the underlying network elements, and therefore can be used to control the array of network equipments from different services presented in the access network, as well the OLTs from G-PON and NG-PON2. It must be taken into consideration that current PONs already have a form of centralized control, running at the OLT, responsible for ONUs bandwidth allocation, wavelength assignment, and several other management and configuration functions, while also establishing communication between OLT and ONUs, thus limiting the benefits of SDN considerations. However, SDN can still add benefits, since OLTs can become more programmable, enhancing the capabilities offered by the standard OMCI and PLOAM functions (which are hardcoded by the vendors), flow mapping rules/policy modifications can be applied on demand, new sources of revenues are created based on QoS, and OLT control becomes unified and centralized, providing an OLT control plane abstracted from the network equipments.

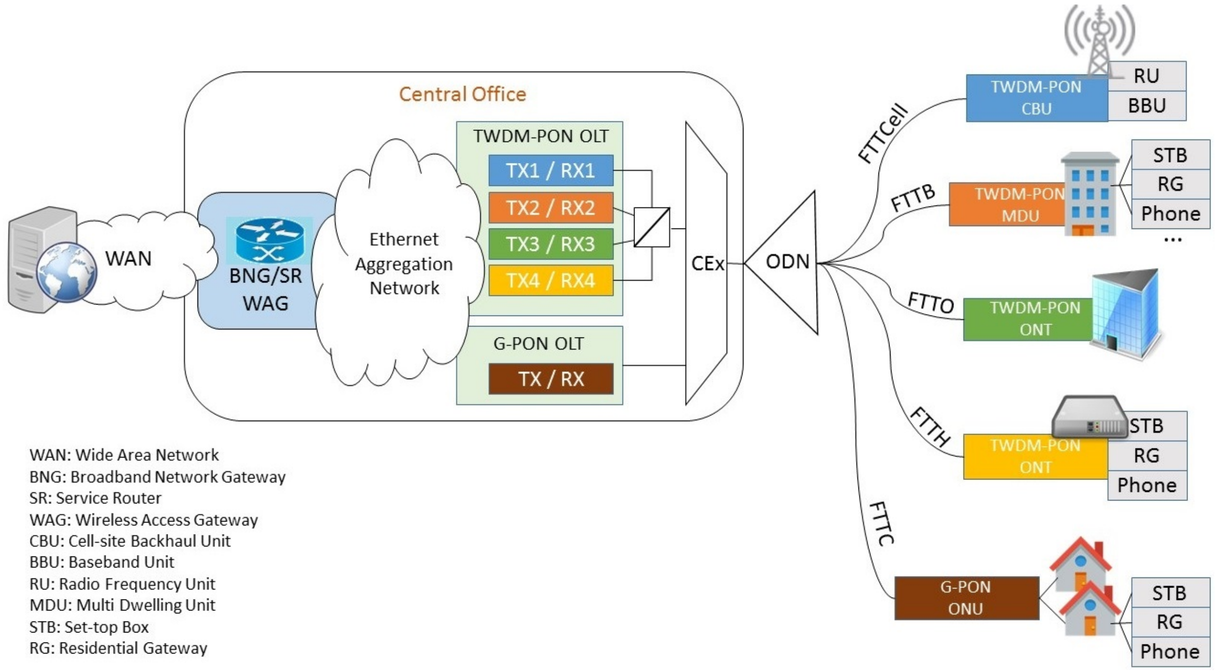


Figure 4.1: Access network infrastructure

The present chapter proposes a controller server architecture that runs on generic hardware, its purpose is to move the network equipments control functions to it, deploying the virtual networks to deliver the desired functions proper behaviour and connecting them to the physical network. It relies on OpenStack software and a top-of-rack OpenFlow switch to provide the controller server infrastructure. The controller server is then used to provide the behaviour of the virtualized hardware functions, that run on it as VNFs, leading to a simplification or complete removal of their hardware counterparts in the network. After explaining the architecture of the controller server, it will be proposed a virtualized OLT running on it, which is chosen as the target for virtualization since it enables a remote and multi-tenant management of the ONUs, from all OLTs connected to the controller, through the OMCI and PLOAM channels, providing a way for different service providers to control their respective ONUs in a shared access infrastructure scenario.

4.2 Proposed Controller Server

4.2.1 Controller Server Architecture

In the access infrastructure seen in figure 4.1, the best solution to include the controller server is the aggregation network since all the traffic has to flow through it, making it ideal to integrate the controller server. In figure 4.2 is depicted the proposed controller server architecture, and at the aggregation network is the top-of-rack OpenFlow switch that is under control by the SDN controller running at the server.

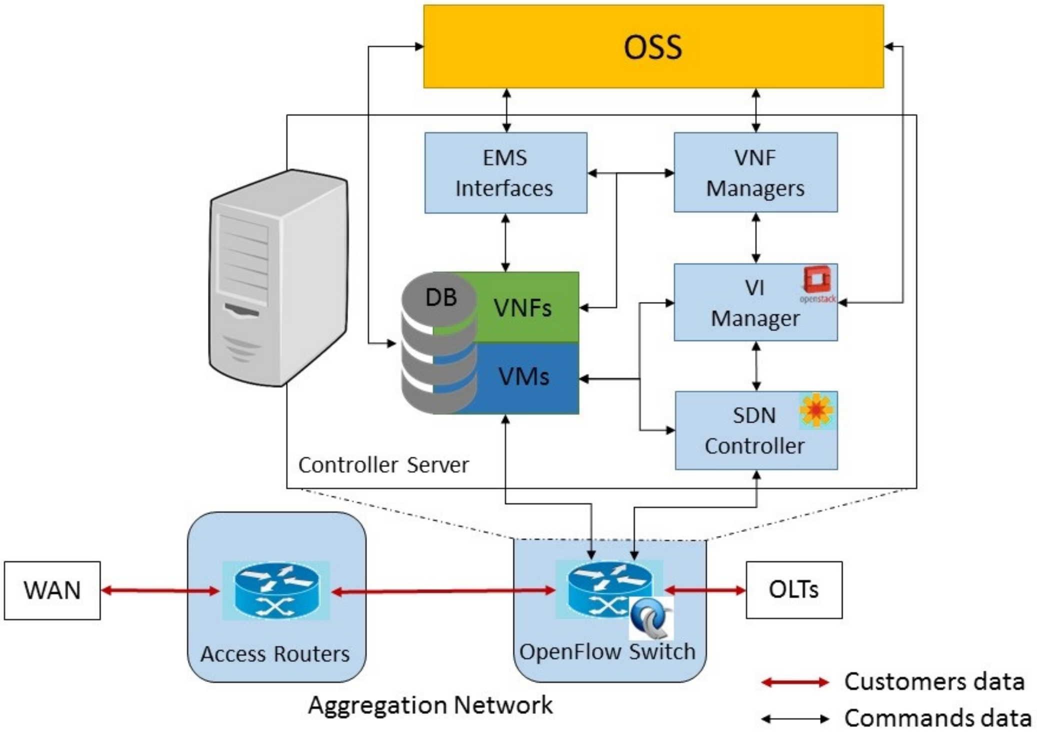


Figure 4.2: Controller server architecture

Below is described the functioning blocks running at the controller:

- Operations Support System (OSS): the controllers intelligence block, responsible for providing an end to end view of the virtualized and non-virtualized network elements. In the proposed controller server it's considered the use of an OSS that also functions as the system orchestrator. For this, it's programmed with intelligent and automatic algorithms to control all the equipments, passing the commands to the EMS interfaces and manager blocks by following the rules programmed into it and respond in real time to requests. For tenants accessing the controller it provides a single and unified view of all the EMS interfaces. The OSS must collect performance and fault statistics from the VMs Database, and events that occur are dealt promptly through the use of pre-defined algorithms. It also communicates to the VNF managers to instantiate VNFs and the VI manager for configuring the network. It provides to the NFV managers the VNFs templates, saving that information in the VNF deployment catalogue. All the tenants that want access to the controller interact with it through applications that provide the services and mechanisms for interaction with the OSS, as well authorization and authentication rules.

- VNF managers: manages the VNFs lifecycle and has a VNF deployment catalogue containing all the configurations and requirements necessary to realize the VNFs functions. Upon receiving instructions from the OSS to initiate one or more VNFs, it consults the catalogue for the VNFs configurations and EMS operational behaviour, and then initiates the VNFs by communicating with the VI manager to request resource reservation and service chains required between them.

- VNFs Database: consists on the set of network virtualized functions, providing an abstraction of the management and implementation details of the virtualized hardware. Each VNF is installed on the VMs instantiated for their functions. As examples there is the Broadband Network Gateway (BNG), OLT or RG functions from the hardware presented in figure 4.1.

- EMS Interfaces: responsible for exposing the deployed VNFs control to the OSS block.

- VI Manager: works by running OpenStack software, and is responsible for managing the VMs Database, elastically adapting them to network and VNF managers demands. Also maps the virtual and physical networks according to requests from the VNF managers and the OSS, communicating to the SDN controller the network configurations.

- VMs Database: contains the abstraction of the virtualized hardware and software, presenting them as VMs to the network. The VNFs use the VMs to provide the pretended behaviour of the non-virtualized hardware. The VMs Database must have enough resources to process VMs functioning, and to store all the VMs deployed. The VMs Database must have elasticity since the VMs needs for resource can float, and future needs might demand the deployment of more VMs.

- SDN controller: it runs on the controller and establishes a connection with the VMs database and the OpenFlow switch for control purposes. It receives from the VI manager the configurations of the networks deployed, that are translated into OpenFlow or OvSDB commands, and configures the virtual and physical networks behaviours according to those commands. The best software solution for its implementation is the OpenDaylight controller [53], which is an open source project by the Linux Foundation. It's chosen since it already has a Driver for the OpenStack Neutron ML2 plugin and supports both OpenFlow v.1.3.0 and OvSDB protocols.

- OpenFlow Switch: is implemented as a top-of-rack OpenFlow Switch to provide higher performance and reliability. Additionally to the OpenFlow protocol it must have OvS capabilities, so an application can monitor and control it externally. The OpenFlow switch aggregates the traffic on the aggregation network and applies rules to manage it through OpenFlow flow tables configured by the SDN controller.

In terms of system performance, the data plane relies on the physical capabilities of the hardware components and transmission rates between them, while the control plane depends on the amount of information processed and stored. Taking these aspects into consideration, the connection between components must provide support to expected traffic peak rates and the OpenFlow switch must have enough ETH ports for all the connections required, while the server hardware that provides the control plane must have the storage and processing capacity to always process information efficiently regardless of traffic flow and number of virtualized elements running on it.

4.2.2 Manager Blocks Software

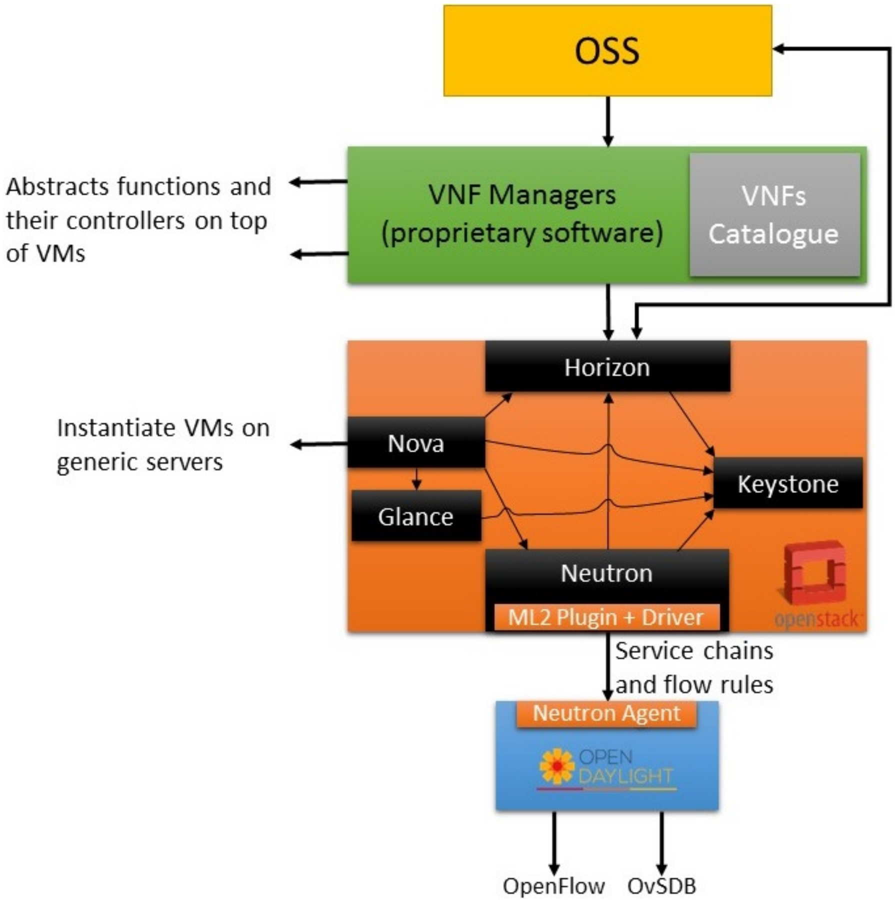

OSS
Abstracts functions and their controllers on top of VMs
VNF Managers (proprietary software)
VNFs Catalogue
Horizon
Instantiate VMs on generic servers
Nova
Keystone
Glance
Neutron
ML2 Plugin + Driver
openstack
Service chains and flow rules
Neutron Agent
OPEN DAYLIGHT
OpenFlow
OvSDB

Figure 4.3: Overview of the manager blocks

The manager blocks are fundamental elements of the controller server, since is the capabilities they provide that enable the deployment of the controller virtual environment. On the controller server there are two manager blocks, as depicted in figure 4.3, that communicate between them to deploy the controller infrastructure, plus the SDN controller that is required to deploy the virtual networks and connecting them to the physical network.

A more detailed explanation is made next on the requirements of each manager block and the communication protocols between them and other functional blocks:

- VNF managers capabilities depend on the services they aim to provide, and thus their software and templates must be created and provided by VNF providers for VNFs and respective EMS interfaces. The VNFs and EMS templates are in the VNF deployment catalogue for the VNF manager to access and deploy each time a set of VNFs is required. The OpenStack components requirements and command flows for VNFs deployment are explicitly provided in the catalogue. The VNF managers are also responsible for the monitoring and scaling of deployed VNFs, that based on their usage and fluctuating service demands might require more resources from the NFVI, communicating these requirements to the OpenStack.

- OpenStack is used as the VI manager of the controller server, providing the NFVI by deploying the VMs and virtual networks needed by the VNFs. On the controller server it's required from the OpenStack framework the following components:
  - Keystone: provides authorization and authentication policies of users communicating with OpenStack components, which in the case of the proposed controller can either be the VNF managers or the OSS. For each user it's necessary to establish their credentials, identifying the set of operations each one can perform. Then a token is used for authenticating requests once a VNF manager or OSS credentials have been verified.
  - Horizon: provides to the users the interfaces for communicating with the Neutron and Nova components.
  - Nova: retrieves from the Glance the VMs images on which the VNFs run, attaching then the information on them based on the VNFs needs and installing them in the Database.
  - Glance: stores VMs Images metadata, providing them to the Nova when requested.
  - Neutron: enforces the service chains and flow rules onto the OpenFlow switch, enabling the OSS to create their own virtual environments by connecting a series of VMs. It's installed with the ML2 + OpenDaylight Driver for communicating with the SDN controller running the OpenDaylight controller.

The proposed controller server can be seen as a generic controller, since the OpenStack components used provide the necessary capabilities to deploy a NFVI capable of running virtualized functions of a vast array of network equipments. It is supposed that in the future the methods to provide NFVI will be well established, and the main focus for revenue is the VNFs and OSS systems that use the NFV architecture to provide network control applications.

VM Instantiation Workflow

One thing in common for every service to be provided is the necessity of creating VM instances on the VM database and add their ports to the network. The whole process is done by the OpenStack components after a request is made by a NFV manager. In figure 4.4 is depicted the workflow of OpenStack requests for VM instantiation.

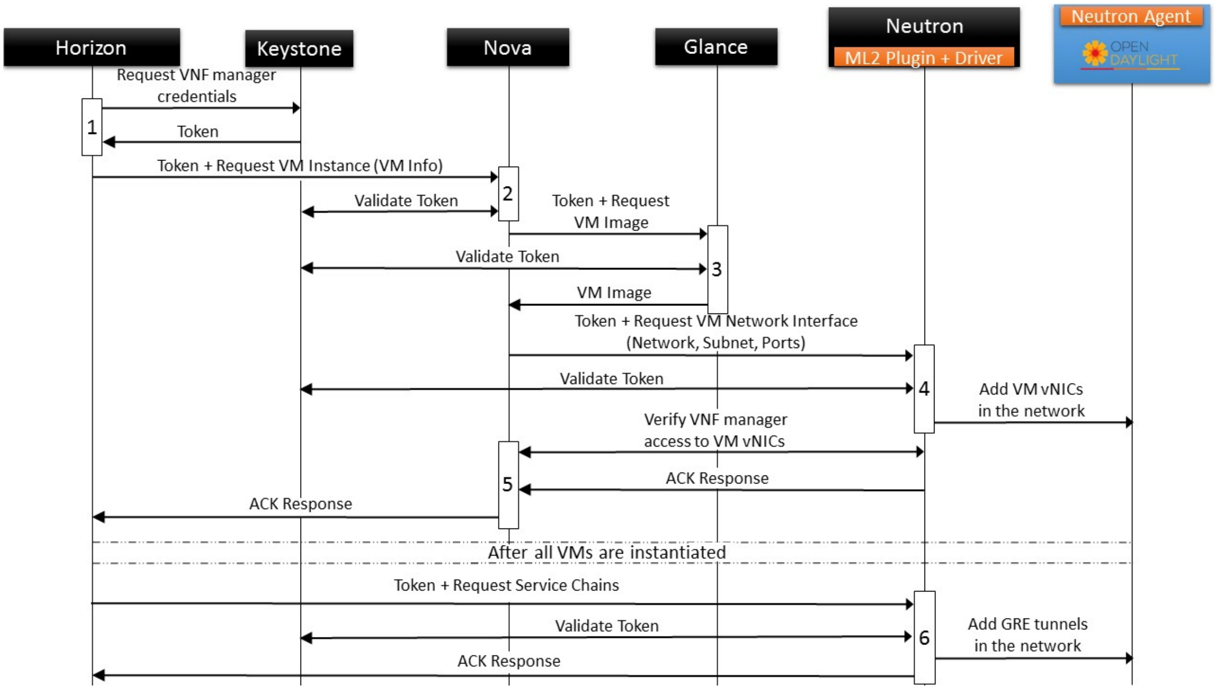

Horizon
Keystone
Nova
Glance
Neutron
ML2 Plugin + Driver
Neutron Agent
OPEN DAYLIGHT
Request VNF manager credentials
1
Token
Token + Request VM Instance (VM Info)
2
Validate Token
Token + Request VM Image
Validate Token
3
VM Image
Token + Request VM Network Interface (Network, Subnet, Ports)
Validate Token
4
Add VM vNICs in the network
Verify VNF manager access to VM vNICs
5
ACK Response
ACK Response
After all VMs are instantiated
Token + Request Service Chains
Validate Token
6
Add GRE tunnels in the network
ACK Response

Figure 4.4: VM instantiation and service chain deployment workflow with OpenStack

- Step 1: when a VNF manager access the Horizon dashboard for a request it first provides its credentials to the Keystone Identity for authentication, and receives a token used for sending requests to other components. The token travels between components accompanying every command within the OpenStack, and every time it reaches a component it is validated by the Keystone before passing the request to the component.

- Step 2: with the token associated to the VNF manager or OSS authorization rules, the Horizon specifies a 'launch VM instance' request to the Nova. The Nova sends the request for token validation and access permission to the Keystone. After receiving the validated token, the Nova interacts with the database to reserves an entry for the VM and saves the VM information (RAM, CPU, Disk space) to be passed to the VM image.

- Step 3: Nova request the VM Image from the Glance by passing to it the token. The token is then validated by the Keystone and the VM Image metadata is delivered to the Nova. The Nova initiates the VM image with it's information on the reserved database entry.

- Step 4: then the Nova requests the VM vNICs to the Neutron by passing to it the token, and Neutron passes the token to the Keystone to be validated. The Neutron then allocates and configures the network by interfacing with the OpenDaylight controller and adds the VM vNICs in the network.

- Step 5: after that, the Neutron and the Nova verify if the VNF manager or OSS have access to the network and VM instance, and if everything is working as it should a successfully ACK response is sent to the VNF manager through the Horizon and the VM instantiation process is ended. On the VNF deployment catalogue is instructed how many VMs are required to provide a service, which can require one or more VNFs running on more than one VM. Steps 1 through 5 are repeated for the required number of VMs. When all VMs are instantiated the VNF manager install the VNFs into the VM reserved for it.

- Step 6: After all the required VMs are instantiated and the VNFs installed in them, the VNF manager requests the service chains as described in the VNF template, communicating to the Neutron, through the Horizon, the port mapping of the service chains to provide, establishing the GRE tunnels between VMs vNICs to provide the required service chains. The process ends with an ACK response sent to the VNF manager. Once all the VNFs are installed successfully with required resources and network, the EMS will configure the VNFs, hosted on the VMs.

4.3 Virtualized Optical Line Termination (vOLT)

Current COs are a combination of OLT equipments, aggregation switches and access routers from different services and providers. With the proposed controller server, network elements can be abstracted, leading to a reduction of the type and quantity of devices and power consumption at the COs. This section proposes a vOLT running on generic servers following the controller server implementation proposed on this chapter.

4.3.1 vOLT Functional Blocks

The study done on the protocol stack for G-PON in section 2.3.3 and for NG-PON2 in section 2.4.4, it's seen that the OMCI and PLOAM functioning blocks work as external clients of the TC layer. These clients work as static software hardcoded into the OLT when it's built, meaning that each vendor configures these clients with the functions they desire to provide, guaranteeing the required minimum functions according to the recommendations. Furthermore, in a multi-tenant access infrastructure, service providers can only manage access routers belonging to their services, so all the management and fault detection capabilities they have are concentrated on these. For extending these capabilities into the ONUs used to deliver their services, they must rely on the network provider to take care of their management. Meaning that for service providers when faults occur or there's a necessity to configure their network equipments they must communicate these to the network provider to take care of them.

The vOLT aims to resolve this questions, enabling a shared management of OLT control, a centralized controller for all OLTs deployed, and an upgradable control plane. For this architecture to be deployed it is needed to simplify the OLT by replacing the traditional OLT hardware with a simple OLT MAC chip with the respective GPON/TWDM-PON TC and PMD layers, and integrating on it an OLT Driver that functions as a OpenFlow switch so it can be under the SDN controller control, responsible for controlling OLT ETH ports and thus enabling traffic aggregation and/or forwarding from/to the traffic ETH, OMCI and PLOAM ports at the OLT MAC. The PLOAM and OMCI ports must have IP address set up to each one to enable the connection between them and the vOLT VNFs. The OMCI and PLOAM clients are created as VNFs that run on the vOLT. In figure 4.5 is depicted the alterations that the OLT MAC will suffer.

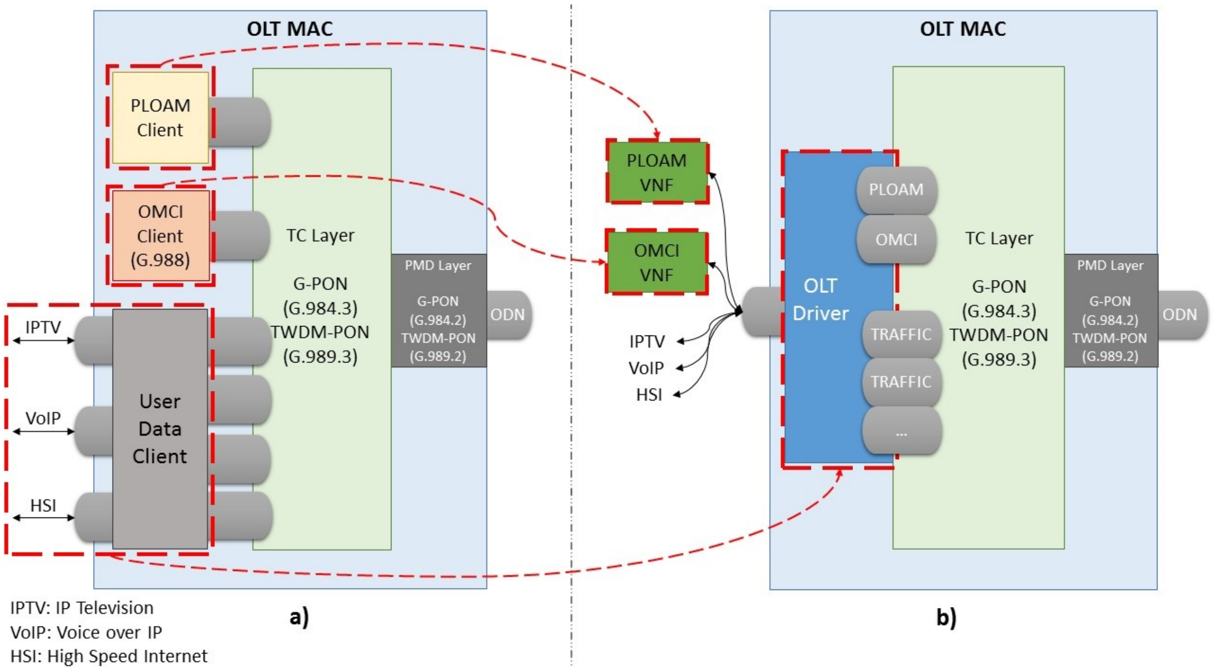


Figure 4.5: Functional blocks of a) traditional OLT and b) simplified OLT with VNFs

The proposed controller server architecture from section 4.2 makes it possible for the vOLT to run a variable number of G-PON and NG-PON2 OLTs control planes, creating a vOLT instance for each OLT connected to the controller server. Then the vOLT controller brings the OLT Drivers under its control by establishing the TLS or TCP connections to them, enabling the management of their OpenFlow flow tables and connecting its VNFs with the OMCI and PLOAM ports, providing to the network provider and their tenants a management interface able to control all OLTs.

vOLT VNFs and OLT Driver Description

The vOLT VNF Manager must provide the VNFs with the behaviour of their non-virtualized counterparts. The templates and software for different VNFs are stored on the VNF deployment catalogue, and once their deployment is requested the VNF manager requests the VI manager to instantiate the VMs necessary to install their software and establish the networks for their communication. After the VNFs software is installed onto the respective VMs and the service chains established, following the process described in section 4.2.2, they are ready to provide their functioning to the OLT MAC.

The VNFs software and OLT Drivers must provide the typical functioning of the non-virtualized clients, but virtualization enables the introduction of enhancements on these. The VNFs and OLT Driver functions are described below:

- PLOAM VNF: its implementation follows the G-PON ITU-T G.984.3 and NG-PON2 ITU-T G.989.3 recommendations, and due to their different requirements, the VNF catalogue must contain a different template for each one and installs the one required for the connected OLT. The PLOAM VNF then enables tenants manual intervention and remote control of PLOAM associated functions described in section 2.5.2.

- OMCI VNF: its implemented as described in the ITU-T G.988 and offers the functionalities described in section 2.5.3, deploying the master entities that offer control over the slave entities at the ONUs. The OMCI client has a set of hardcoded mandatory and other optional managed entities when the OLT hardware is built. OMCI client virtualization enables the deployment of new managed entities at anytime by updating the VNF software. Since the VNF runs at the server, it enables ONUs management and configurations to be done from the tenants facilities, enabling it to make adjustments remotely and response quicker to alarm signalling that can't be solved by the vOLT. Also related with the OMCI functions is the MIB, that can be deployed as a different VNF, and every time OMCI data flows between the OMCI VNF and the OLT MAC, the OMCI VNF updates the MIB, and can be accessed by the tenants for configuration, management and performance statistics information.

- OLT Driver: aims to provide the functions done by the user data client but with enhancements and the capability of being manipulated. It takes responsibility of prioritizing packets according to their S-VLAN in the Q-in-Q packet format for traffic flowing out/in the OLT, offering the benefit of removing from the OLT the need for specific ports to deliver the different types of services traffic to their correct access routers (access routers traffic distribution is passed to the OpenFlow switch at the vOLT, which provides more ports to connect to different access routers, leading to a reduction of aggregation switches in the network and thus a simplified management of COs), and routing OMCI and PLOAM traffic from/to the correct OLT port. Due to being deployed has a OpenFlow switch it enables a further traffic differentiation, by applying OpenFlow flow tables match rules it can further differentiate traffic by L2 to L4 headers, enabling a better differentiation of clients.

4.3.2 vOLT Traffic Flow

Study done on the OpenStack in section 3.4 provides a brief introduction on how the network works using the Neutron component. In the proposed vOLT all the VNFs for a vOLT instance run on the same compute node as a way to facilitate their deployment and traffic flow. The nodes required to understand and the traffic flow between the PLOAM and OMCI VNFs with their OLT ports are explained in the following section.

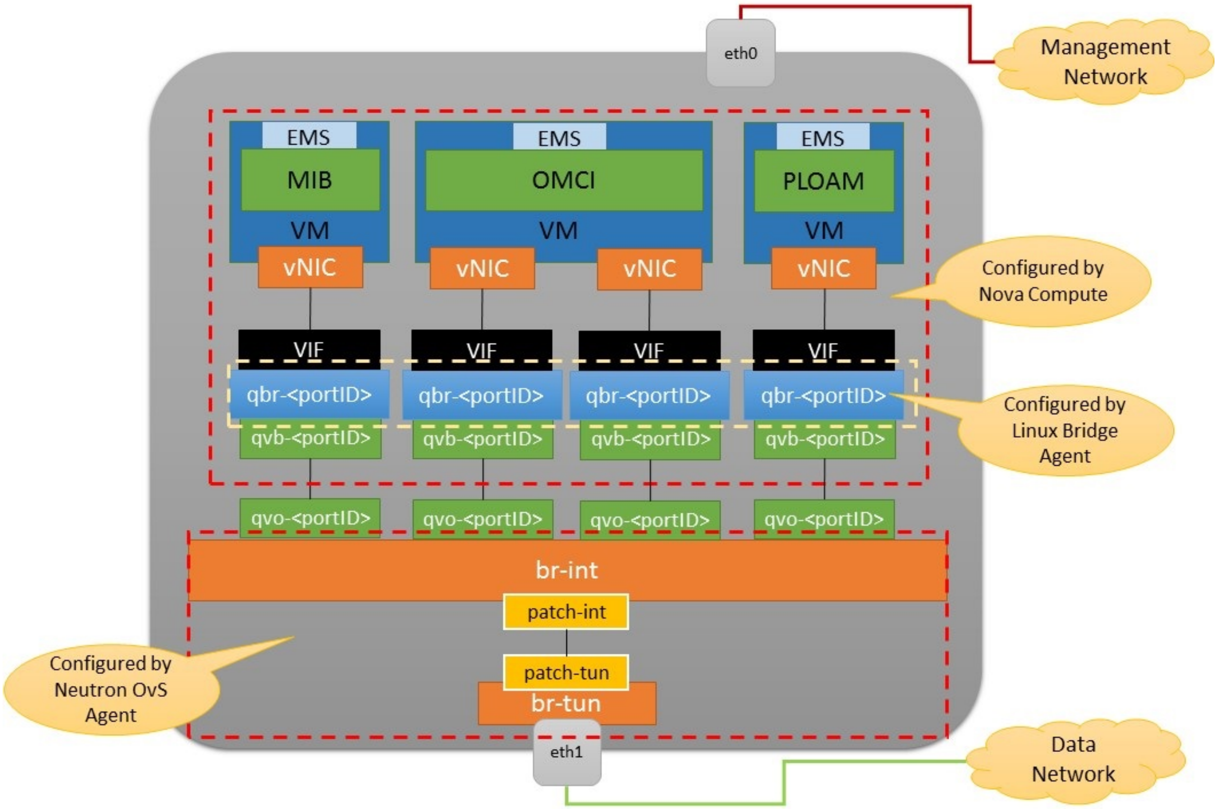


Figure 4.6: Compute node configuration

Figure 4.6 provides a comprehensible depiction of a compute node, which contains a Neutron OvS agent used to deploy the OvS bridges within the node and the GRE tunnels that provide connectivity between nodes, the Nova compute agent that deploys the VMs running on the node, and the Linux Bridge agent that deploys the Linux bridges required for the connectivity between VMs and the network. On figure 4.7 is provided the depiction of the network node (the figure contains an example for two vOLT instances), containing the OvS agent, a Neutron L3 agent that provides connectivity to the external network, and also a DHCP agent.

The network node, configured as seen in figure 4.7, has a **dnsmasq** for each vOLT instance, that are deployed by the DHCP agent, providing IP addresses for each instance VMs, and also a **qrouter**, that has a **qr-** virtual port (each OvS port has an associated IP address) connecting to the **Integration Bridge (br-int)** for each vOLT instance. It also has at least three network ports, **eth0**, **eth1** and **eth2**. The **eth2** port connects to the OpenFlow switch, enabling the VNFs to establish a connection with the OMCI and PLOAM ports.

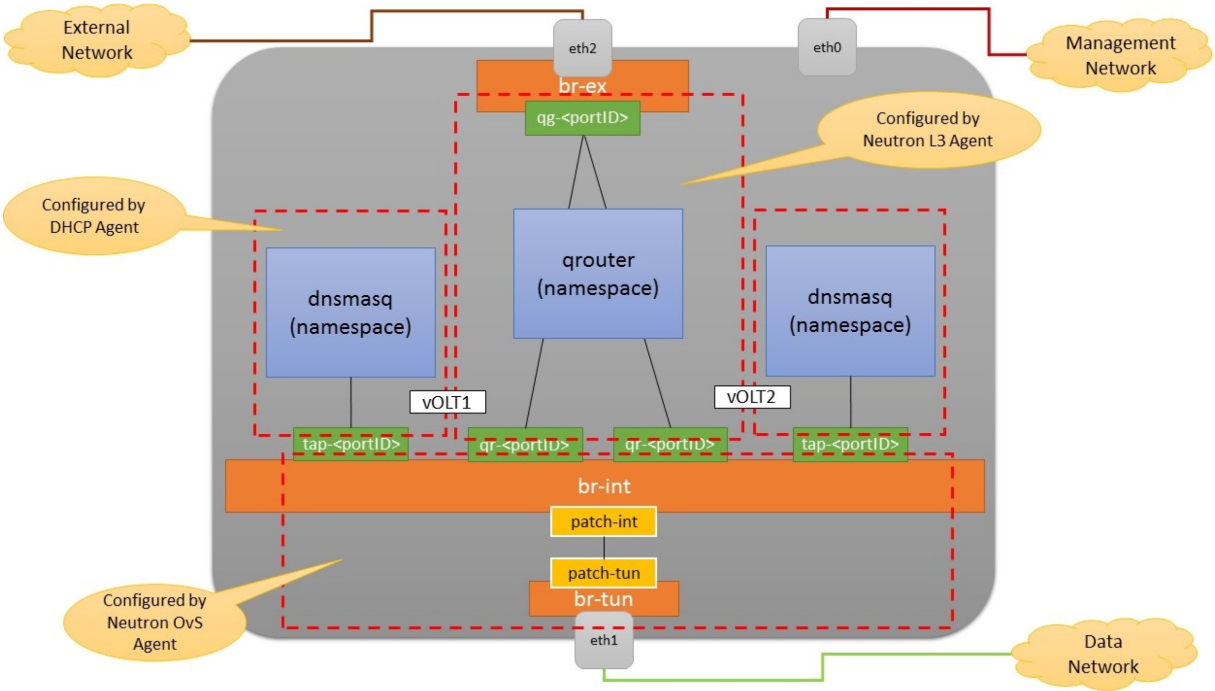


Figure 4.7: Network node configuration

The traffic flow steps that packets undergo is depicted in figure 4.8 and explained in detail below. For this explanation the first packet is sent by the VM, the reason is that the OMCI and PLOAM ports don't know where to send their packets when they are initiated, and thus the first packet must be sent from the OMCI and PLOAM VNFs to the clients IP ports. This process must be done after the vOLT instance for the respective OLT is instantiated, configuring in them the IPs to which they must communicate, and thus enabling a back and forth communication of control packets. The PLOAM and OMCI ports when receive the packets for the first time know that the source IP of the packets is to where they must send their packets.

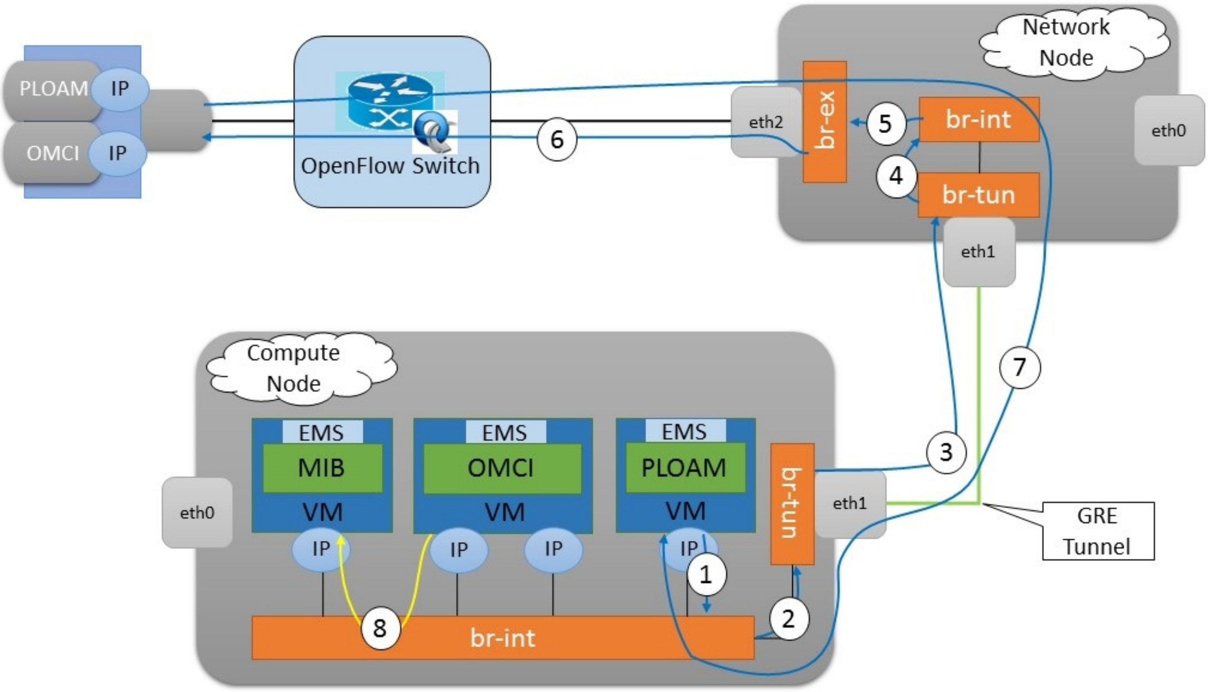


Figure 4.8: vOLT traffic flow

- Step 1: The VNF sends the packet with the destination IP address of the respective OLT and client port, the packet goes with the VM **vNIC** internal IP as source IP. The packet then must be sent to **br-int**, but the connection between VMs and the **br-int qvo-** ports cannot be deployed as a direct connection due to OvS bridge capabilities. The problem is that OvS bridges can't directly attach a network TAP device, used to provide the **Virtual Interfaces (VIFs)** where iptables [54] rules are applied. To solve this problem it's required that the ML2 Linux Bridge agent deploys a Linux bridge for all VMs on all compute nodes, providing the required L2 connectivity between VMs and the external network, as well security groups management. With the Linux bridge deployed, the VM forwards the packet to the **VIF** connected with the **qbr-** of the Linux Bridge, and then it connects with the **br-int qvo-** port through the **qvb-** port.

- Step 2: The packet is received by **br-int** and is tagged with a VLAN tag associated with the sender VM that represents the virtual network it belongs (meaning a VLAN per vOLT instance). The **br-int** must now decide where to forward the packet, so the **br-int** uses the Address Resolution Protocol (ARP) to find the destination of the packet (ARP is only used when a packet doesn't already have a register in the ARP cache for it's destination IP), and with the packet tagged with an external IP it receives a response from the external network, so **br-int** knows it must send it to the network node. The packet is then sent to **Tunneling Bridge (br-tun)** through **patch-int**, that pairs with **patch-tun**, functioning as trunk ports [55].

- Step 3: The **br-tun** associates the VLAN tag of the packet to GRE tunnel IDs configured by the OpenDaylight controller relying on OpenFlow actions, and then encapsulates it with a GRE header containing also a GRE tunnel key. The now GRE packet is sent through the GRE port specified by the GRE tunnel key, exiting the compute node via **eth1** port. The GRE packet flows across the data network through the used GRE tunnel and arrive at the **eth1** port of the network node.

- Step 4: The GRE packet, before being received by the network node **br-tun**, have its GRE tunnel key matched with the GRE port used, making the GRE packet enter the **br-tun**. At the **br-tun** the GRE packet is matched by OpenFlow flow tables, and de-encapsulating actions are applied to retrieve the original packet, forwarding it to **br-int** through the **patch-tun** to **patch-int** pair (trunk ports).

- Step 5: The **br-int** receives the packet knowing the VLAN it belongs, and then must forward it to the **qrouter qr-** port corresponding to the packet VLAN tag through OpenFlow actions. The **qrouter** sets up iptables and NAT rules according to the routing tables, virtual networks and floating IPs created. Before the packet goes to the external network it's applied to it Source NAT (SNAT) to replace the VM source internal IP with its floating IP. The packet going out of the **qrouter** is now NAT translated with the VM floating source IP, enabling it to be identified in the public network, for the OMCI and PLOAM ports to know where to respond. The packet is then sent to the **External Bridge (br-ex)** through the **qg-** port.

- Step 6: The packet is received by the **br-ex**, which is deployed to provided the connection between the **qrouter** and the **eth2** port. The **eth2** sends the packet to the external network, which passes through the OpenFlow switch and arrives at the target OMCI or PLOAM ports.

- Step 7: With the connection established with the VMs, packets can now flow knowing the network path to use. When packets flow from the OMCI or PLOAM to the VMs, the traffic flow steps is the same as explained above with a minor alteration at the network node. This alteration is due to the sent packet having its destination IP as the VM floating IP, so when it arrives at the **qrouter** it applies Destination NAT (DNAT) and replaces the packet destination IP with the VMs internal IP. Also to note that the first packet from each of the ports to arrive at the network node **br-int** triggers it to use the ARP so it updates its ARP cache and forwards the packet to the correct VM.

- Step 8: This step is only applied for the communication between the OMCI and the MIB VNFs. In the vOLT deployment the OMCI VNF is deployed with two **vNIC** ports, one has the purpose of communicating with the external network, and the other for communicating with the MIB VNF. When the OMCI sends a packet to the MIB VM IP, it forwards it to the **br-int**. The **br-int** uses ARP to discover the MIB IP location, which is running on the same compute node, so the packet is forwarded there through the respective **qvo-** port.

The packets flowing between the clients VNF to the clients ports transport in their data field the message to be communicated to the OLT. On the typical behaviour of the clients these messages are directly sent to the TC layer to be encapsulated into the GEM/XGEM frames, this requires that the OLT must be capable of de-encapsulating the incoming packets to retrieve the control message and only after doing this it can send the message to the OLT TC layer, so it can be sent to the ONUs. On the inverse direction the same must occur, so the ONU messages can be processed by the clients VNFs.

4.3.3 vOLT Instance Creation

When an OLT is installed on a CO with a vOLT running on it, and is started for the first time, it's required to instantiate and configure the vOLT instance for it. The first step is for the OLT Driver OpenFlow switch to establish the independent connection with the OpenDaylight controller, bringing the OLT Driver to the control of the vOLT and permitting the establishment of the to be deployed control VNFs with the OLT control interfaces. The workflow that is then performed by the vOLT software to instantiate a vOLT instance for the new OLT is show in figure 4.9.

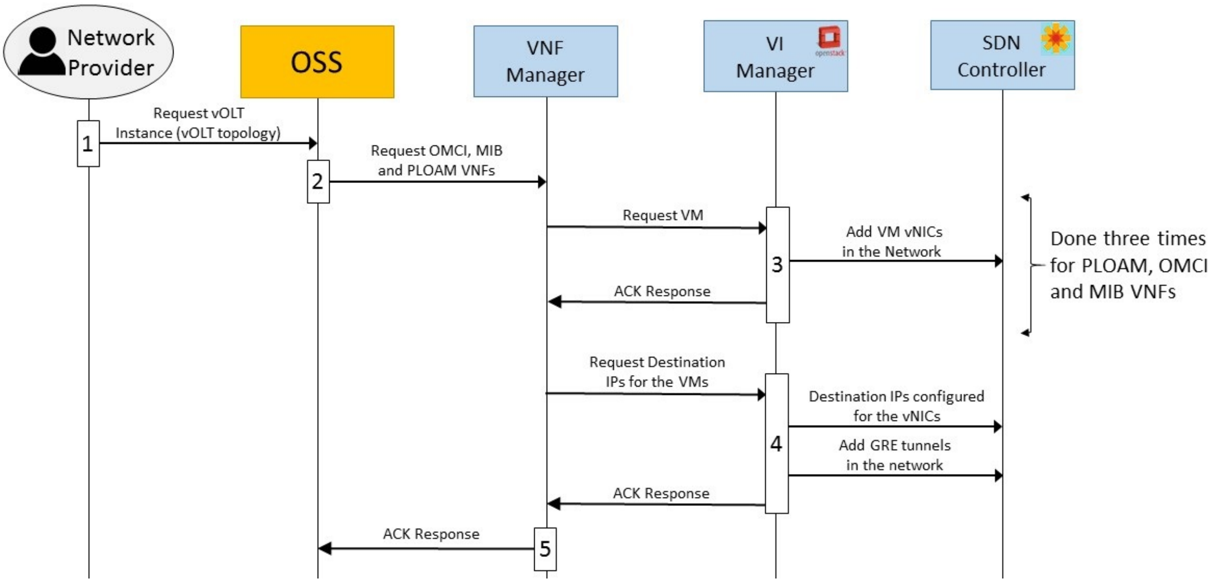

Network Provider
OSS
VNF Manager
VI Manager
SDN Controller
1
Request vOLT Instance (vOLT topology)
2
Request OMCI, MIB and PLOAM VNFs
Request VM
3
Add VM vNICs in the Network
ACK Response
Done three times for PLOAM, OMCI and MIB VNFs
Request Destination IPs for the VMs
4
Destination IPs configured for the vNICs
Add GRE tunnels in the network
ACK Response
ACK Response
5

Figure 4.9: vOLT instance creation workflow

- Step 1: The OSS receives a request from the vOLT administrator to create a vOLT instance for the newly attached OLT, in the request is defined the OLT topology (G-PON or TWDM-PON system) and IPs for the clients ports at the OLT. The OSS must be able to apply security measures on requests done to it, so it can confirm the identity of the user making the request.

- Step 2: The OSS requests the VNF manager to instantiate the OMCI, MIB and required PLOAM VNFs. It also communicates to it the destination IPs for the OMCI and PLOAM VNFs.

- Step 3: The VNF manager checks the requested VNFs in the VNF deployment catalogue, communicating the requests to the VI manager to instantiate the VMs on the VMs Database according to the VNF templates. The process on section 4.2.2 (except step 6) is done three times, one for each VNF to be instantiated, with the configurations of each VM. Each time one VM is instantiated the VNF manager receives an ACK response to confirm the correct functioning of the VM and it's vNICs can communicate with the VNF manager, installing the VNF software on the VM.

- Step 4: With all the VMs instantiated and the VNF software running on them, the VNF manager requests the configuration of the destination IP addresses on the VMs vNICs, described below:

  - PLOAM VNF destination IP to PLOAM client IP port.
  - OMCI VNF has two vNICs, the OMCI VNF software must be made to have one just to update the MIB and another to establish the communication channel with the OMCI port. Knowing this, the VNF manager associates the OMCI VNF ports with the correct destinations, one port has the destination IP of the MIB VNF vNIC and the other has the destination IP of the OMCI client IP port.

  The VI manager then requests the SDN controller to configure the requested Destination IPs for the VMs vNICs and the deployment of the GRE tunnels between the compute node and the network node. The process ends with an ACK response to the VNF manager confirming the end of the process.

- Step 5: The VNF manager confirms to the OSS the end of the process and the vOLT instance is now created.

After the vOLT instance is created the OSS must request to the VNFs to send a first packet to their destinations for the communication channel to be established between them and their clients port. With this process done the OSS of the vOLT sends the required commands to the OLT to initialize the OLT MAC and set up all the attributes with their default values.

4.3.4 vOLT Use-Case

The virtualization of OLT functions can be of extreme value to network providers and their tenants. From network providers perspective it can reduce CAPEX, by deploying simpler COs, and OPEX by introducing the capability of passing some OLT functions to their tenants and automatizing others by deploying an intelligent OSS capable of controlling all the aspects of vOLT deployment and control, reducing the need for interacting directly with the OLT. Figure 4.10 shows the use-case architecture of the vOLT. The VNF manager is excluded of the use-case since its functions are only applied for VNFs management, and not the control of the network and its elements.

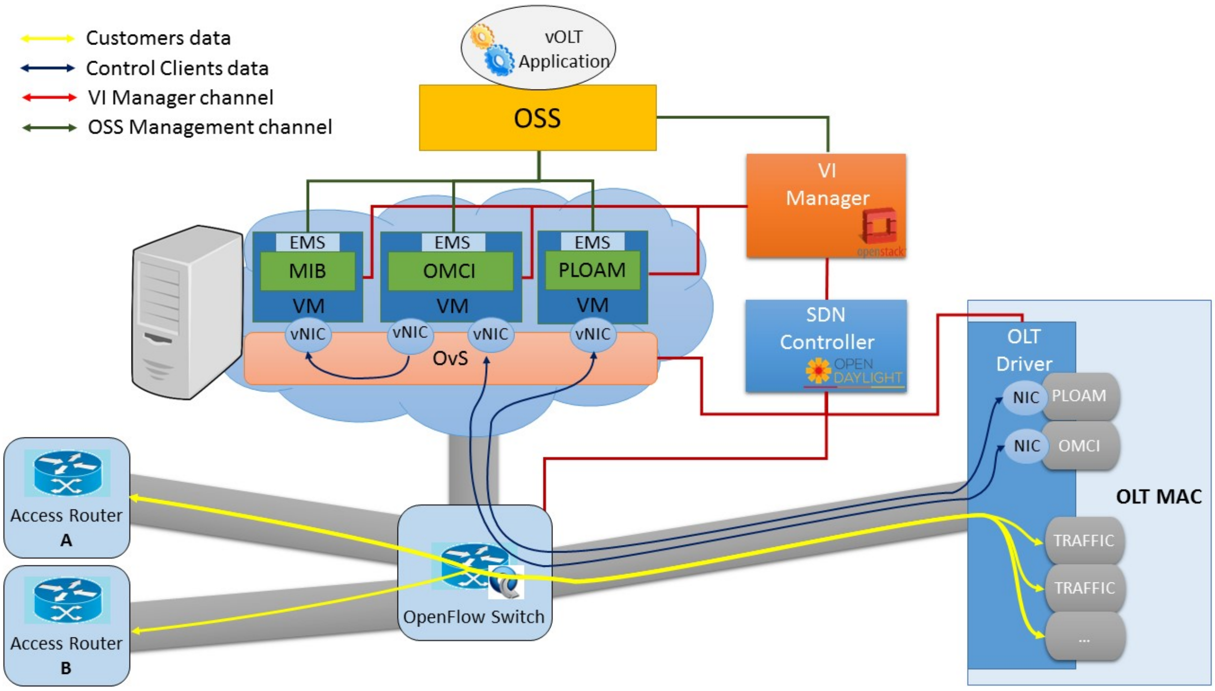


Figure 4.10: vOLT use-case architecture

The use-case shown relies on a multi-tenant vOLT application that should be aimed to work as a web portal for service providers to manage their ONUs. In it should be defined sessions for the different tenants so authentication and authorization rules can be delineated on the access to functions. To do this it's required the deployment of a vOLT application that abstracts and simplify the functions provided, not showing directly the OMCI and PLOAM commands but providing an user-friendly interface with simple commands, translating those into the commands required to be introduced in the EMS interfaces of the VNFs. The vOLT application interacts with the OSS that is responsible for the control of VNFs through their EMS interfaces, that on receiving the commands from the vOLT application takes the task of introducing them into the OMCI or PLOAM VNFs. Also, the vOLT application must provide real time information and statistics about the system when requested, and this is done by requesting to the OSS to fetch that information from the MIB VNF and provide it in a simple manner to tenants.

Provide ONU Management to Tenants

To provide ONU management to tenants it's required to first associate an ONU with the tenant it belongs. This process can be done by making the vOLT application ask the tenants to communicate the serial number of their ONU equipment. Then the vOLT application sends a request to the OSS to check on the MIB VNF the ONU-ID associated with that serial number, and when that information arrives at the vOLT application it associates that ONU-ID to that tenant session.

With the ONU associated to its tenant a vast array of ONU management capabilities can be provided. One example is the capability for ONUs Alloc/Port-ID management, which access to it enables tenants to better differentiate customers based on their needs, instead of having static configurations per ONU, these can be manipulated at anytime from the tenants facilities without the need of the network provider. One benefits introduced by this is if a customer desires a type of connection for live streaming, gaming, or file seeding, where upstream transmission is more demanding and wants to achieve the best possible QoS, it can be assigned to him more Alloc-IDs, thus giving more upstream opportunities to his ONU.

Furthermore, the vOLT application must be able to collect data periodically from the MIB VNF and retrieve it to tenants session. This must be an essential service provided by the vOLT application since alarms and performance checking are essential for tenants to maintain customers satisfaction and QoS by targeting the problems found in real-time instead of being communicated by the customer.

Aggregation Network Simplification and Management

The vOLT deployment as show in figure 4.10 offers the enhancement of providing a aggregation network fully programmable due to the deployment of the top-of-rack OpenFlow switch in its conception and the use of OLT Drivers working as OpenFlow switches, and all of these under the control of the SDN controller, and thus the OSS. This service should be deployed as a different service running on top the OSS, and leads to a aggregation network shared by different service providers and operators, even with the scaling of their services. In resume, in the same aggregation network there can be deployed a new access router without conflicting with others already deployed and without the need of adding more switches into the network, these are directly connected to the OpenFlow switch that is then programmed with the new set of rules into its flow tables.

In section 3.2 is detailed all the set of match field and actions that can be applied in the switches. On the aggregation network a required set of actions it's to define the priority of packets flowing in the network according to their S-VLAN tag. The OpenFlow switch v.1.3.0 is ideal for this since it supports flow tables pipeline, so the first flow table should always prioritize first the control packets flowing from/to the vOLT controller and the respective OLT Drivers and then the packets S-VLAN ID, which is the outer-most VLAN tag on packets, before passing them to the next flow table that distributes the packets to their correct destinations.

Chapter 5

Conclusions

The study done on the current dissertation shows that the emergence of the NG-PON2 is a natural evolution on access networks, it utilizes the TWDM-PON technology so it can meet the current traffic and bandwidth demands due to the increasing number of customers and applications that require access to the network. Furthermore, the TWDM-PON technology enables the co-existence with current legacy PON technologies on the same ODN, and thus protects the investment done by network providers on current infrastructures. To bring further innovation onto the networks it was studied the SDN and NFV architectures, which aim to abstract the control/management plane of network equipments and provide their functioning on dedicated data centers, where northbound APIs can be made to better control their characteristics. By abstracting the network functions, the service providers can make their own applications, enabling them to better manage hardware resources and deliver customized services.

The proposed controller architecture made on this dissertation aims to use the advantages that SDN and NFV architectures introduce and apply them onto the access network. First it was proposed a controller server architecture using the NFV architecture to provide a virtualized environment relying on OpenStack software. The characteristics and the software required for its functional blocks were delineated.

The main proposal of this dissertation is a virtualized OLT implementation, so the benefits brought by the NFV can be integrated on the management of the access networks. This solution relies on the proposed controller architecture, which serves as the backbone of the vOLT, aiming to abstract the PLOAM and OMCI clients from the OLT hardware, and was also described a method for aggregating the traffic from all OLT ports by embedding an OpenFlow switch onto the OLT. Some use-cases and benefits that can be provided by the vOLT are explained so it can serve as motivation for continuing the research on the topic. The vOLT is described with all the necessary requirements for the VNFs and the explanation of the network that supports it, so the next step its to effectively test its implementation.

5.1 Future Work

The first point for future work for the vOLT is the deployment of the VNFs of the OMCI and PLOAM clients of the G-PON and TWDM-PON OLTs. Once these are provided it is required to effectively test the proposed vOLT and understand its behaviour, checking for faults and strengths it has.

One crucial point that requires a better solution is the necessity of providing a public ID to the OLTs PLOAM and OMCI ports, and a possible solution requires that the interfaces are extended to be under the Neutron network, so they are provided with fixed IPs from the Neutron network and thus can be addressed by the VMs through GRE tunnels directed to their ports.

Another possible function that can be provided in the future by the vOLT is the Deep Packet Inspection (DPI) functionality on packets flowing through the network, searching for protocol non-compliance, viruses, spam, intrusions, or for collecting statistical information. On a first attempt of this dissertation this was tried to be implemented as a VNF on the vOLT but due to NAT incompatibilities it couldn't be implemented, the solution for this requires a full virtualization of the network so packets flow through it encapsulated into GRE or VxLAN packets so they always conserve their original headers and thus these aren't altered by NAT when entering and leaving the vOLT.

One last point for future work is the necessity to create an OSS software that enables an automated and intelligent management of OLT functions.

Bibliography

[1] Naoto Yoshimoto. NTTs Access Network Vision towards 2020. In *Software-Defined Optical Access: Hope or Hype?* NTT corporation, March 10 2014.

[2] Leonid G. Kazovsky, Ning Cheng, Wei-Tao Shaw, David Gutierrez, and Shing-Wa Wong. *Broadband Optical Access Networks.* Wiley, 2011.

[3] Rajiv Ramaswami, Kumar Sivarajan, and Galen Sasaki. *Optical Networks: A Practical Perspective.* Morgan Kaufmann, November 27 2009.

[4] OFCOM. Fiber capacity limitations in access networks. Technical report, Analysis Mason, January 13 2010.

[5] MULTICOM. Multicom's G-PON Solution. `http://www.multicominc.com/product-category/solutions/video-data-voice/fiber-optics/gpon/`, 2014 [accessed October 3, 2014].

[6] Recommendation ITU-T G.984.1 (2008), Gigabit-capable passive optical networks (G-PON): General characteristics, March 2008.

[7] Recommendation ITU-T G.984.2 (2003), Gigabit-capable Passive Optical Networks (G-PON): Physical Media Dependent (PMD) layer specification, March 2003.

[8] Recommendation ITU-T G.984.5 (2007), Gigabit-capable Passive Optical Networks (G-PON): Enhancement band, September 2007.

[9] ANACOM. FTTH/B/P networks. `http://www.anacom.pt/render.jsp?categoryId=340669#.VWiHUM9Viis`, January 8 2011 [accessed October 16, 2014].

[10] Recommendation ITU-T G.984.3 (2014), Gigabit-capable Passive Optical Networks (G-PON): Transmission convergence layer specification, January 2014.

[11] Huawei Technologies Co. Ltd. GPON-Fundamentals - Technical Team from FTTH Marketing Department. `http://pt.slideshare.net/mansoor_gr8/gpon-fundamentals`, August 18 2011 [accessed October 16, 2014].

[12] Rajiv Ranjan. GPON. `http://pt.scribd.com/doc/122124416/GPON#scribd`, January 25 2013 [accessed October 22, 2014].

[13] Onn Haran and Amir Sheffer. The Importance of Dynamic Bandwidth Allocation in GPON Networks. White paper, PMC-Sierra, Inc., January 2008.

[14] Recommendation ITU-T G.989.1 (2013), 40-Gigabit-capable passive optical networks (NG-PON2): General requirements, March 2013.

[15] Recommendation ITU-T G.989.2 (2014), 40-Gigabit-capable passive optical networks 2 (NG-PON2): Physical media dependent (PMD) layer specification, December 2014.

[16] Romain Brenot, Ed Harstead, Ron Heron, Thomas Pfeiffer, Wolfgang Poehlmann, Joe Smith, and Dora van Veen. Next Generation Optical Access Technologies. Tutorial, Peter Vetter - Bell Labs, Alcatel-Lucent, September 18 2012.

[17] Zhengxuan Li, Lilin Yi, and Weisheng Hu. Key technologies and system proposals of TWDM-PON. *Frontiers of Optoelectronics*, 6(1):46–56, March 2013.

[18] Ning Cheng, Jianhe Gao, Chengzhi Xu, Bo Gao, Dekun Liu, Lei Wang, Xuming Wu, Xiaoping Zhou, Huafeng Lina, and Frank Effenberger. *Optics Express - Flexible TWDM PON system with pluggable optical transceiver modules.* Optical Society of America, January 24 2014.

[19] Recommendation ITU-T G.987.3 (2014), 10-Gigabit-capable passive optical networks (XG-PON): General requirements, January 2010.

[20] Salem Bindhaiq, Abu Sahmah M. Supa'at, Nadiatulhuda Zulkifli, Abu Bakar Mohammad, Redhwan Q. Shaddad, Mohamed A. Elmagzoub, and Ahmad Faisal. Recent development on time and wavelength-division multiplexed passive optical network (TWDM-PON) for next-generation passive optical network stage 2 (NG-PON2). *Optical Switching and Networking*, June 24 2014.

[21] Recommendation ITU-T G.987.3 (2014), 10-Gigabit-capable passive optical networks (XG-PON): Transmission convergence (TC) layer specification, January 2014.

[22] Denis A. Khotimsky. NG-PON2 Transmission Convergence Layer: A Tutorial. *Journal of Lightwave Technology*, 34(5), March 1 2016.

[23] Recommendation ITU-T G.9802 (2015), Multiple-wavelength passive optical networks (MW-PONs), April 2015.

[24] Recommendation ITU-T G.984.2 (2008), Gigabit-capable Passive Optical Networks (G-PON): Physical Media Dependent (PMD) layer specification - Amendment 2, March 2008.

[25] Hongxiang Wang, Yangyang Liang, Rentao Gu, Yuefeng Ji, Yiran Ma, Chengliang Zhang, and Xiaomu Wang. LP-DWBA: A DWBA algorithm based on linear prediction in TWDM-PON. In *14th International Conference on Optical Communications and Networks (ICOCN)*. IEEE, July 3-5 2015.

[26] Man Soo Han. Dynamic Wavelength and Bandwidth Allocation for Power Saving in TWDM PON. *Advanced Science and Technology Letters*, 54(17-20), 2014.

[27] Recommendation ITU-T G.988 (2012), ONU management and control interface (OMCI) specification, October 2012.

[28] The Open SDN Architecture. Technical report, Big Switch Networks Inc., July 2013.

[29] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. Network Configuration Protocol (NETCONF). `https://tools.ietf.org/html/rfc6241`, June 2011 [accessed October 1, 2015].

[30] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis. The Locator/ID Separation Protocol (LISP). `https://tools.ietf.org/html/rfc6830`, January 2013 [accessed October 1, 2015].

[31] SDNcentral. What are SDN Southbound APIs? `https://www.sdxcentral.com/sdn/resources/southbound-interface-api/`, 2014 [accessed November 3, 2014].

[32] SDN and the Future of Service Provider Networks. White paper, Fujitsu Network Communications Inc., 2014.

[33] SDNcentral. What are SDN Northbound APIs? `https://www.sdxcentral.com/sdn/resources/north-bound-interfaces-api/`, 2014 [accessed November 3, 2014].

[34] Infrastructure SDN with Cariden Technologies. White paper, Cariden Technologies Inc., August 15 2012.

[35] SDNcentral. Whats Software-Defined Networking (SDN)? `https://www.sdxcentral.com/resources/sdn/what-the-definition-of-software-defined-networking-sdn/`, 2014 [accessed November 3, 2014].

[36] Greg Goth. Software-Defined Networking Could Shake Up More than Packets. *Internet Computing, IEEE*, 15(4):6–9, June 30 2011.

[37] Software Defined Networking: The New Norm for Networks. White paper, Open Networking Foundation, April 13 2012.

[38] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. White paper, ACM SIGCOMM Computer Communication Review, April 2008.

[39] OpenFlow Switch Specification, Version 1.3.0 (Wire Protocol 0x04) . Technical report, Open Networking Foundation, June 25 2012.

[40] Ben Pfaff and Bruce Davie. The Open vSwitch Database Management Protocol. Technical report, VMware Inc., December 2013.

[41] Justin Pettit, Ben Pfaff, Martin Casado, Teemu Koponen, Keith Amidon, and Scott Shenker. Extending Networking into the Virtualization Layer. Technical report, October 5 2009.

[42] Justin Pettit, Jesse Gross, Ben Pfaff, Martin Casado, and Simon Crosby. Virtual Switching in an Era of Advanced Edges. Technical report, August 30 2010.

[43] sdx central. What is ETSI ISG NFV. `https://www.sdxcentral.com/resources/nfv/etsi-isg-nfv/`, [accessed February 25, 2015].

[44] ETSI GS NFV 002 v1.1.1 (2013-10): Network Functions Virtualisation (NFV); Architectural Framework, October 2013.

[45] OpenStack Training Guide. Technical report, OpenStack Foundation, May 30 2015.

[46] Lauren Sell. OpenStack Launches as Independent Foundation, Begins Work Protecting, Empowering and Promoting OpenStack. `http://www.businesswire.com/news/home/20120919005997/en/OpenStack-Launches-Independent-Foundation-Begins-Work-Protecting`, September 19 2012.

[47] Atul Jha, Johnson D, Kiran Murari, Murthy Raju, Vivek Cherian, and Yogesh Girikumar. OpenStack Beginner's Guide (for Ubuntu - Precise). Technical report, CSS Corp, May 7 2012.

[48] Sridhar Rao. SDNs Scale Out Effect on OpenStack Neutron. `http://thenewstack.io/sdn-controllers-and-openstack-part1/`, 2015 [accessed June 13, 2016].

[49] OpenStack Operations Guide. Technical report, OpenStack Foundation, June 2016.

[50] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. Generic Routing Encapsulation (GRE). `https://tools.ietf.org/html/rfc2784`, March 2000 [accessed June 19, 2016].

[51] OpenStack Installation Guide for Ubuntu 12.04/14.04 (LTS). Technical report, OpenStack Foundation, June 1 2015.

[52] OpenStack Security Guide. Technical report, OpenStack Foundation, June 2016.

[53] Linux Foundation. ODL Beryllium (Be) - The Fourth Release of OpenDaylight. `https://www.opendaylight.org/odlbe`, [accessed June 14, 2016].

[54] iptables(8) - Linux man page. `http://linux.die.net/man/8/iptables`, [accessed June 20, 2016].

[55] Trunk Port. https://www.techopedia.com/definition/27008/trunk-port, [accessed June 29, 2016].