Small cElls coordinAtion for Multi-tenancy and Edge services

Grant Agreement No.671596

Topic: H2020-2014-ICT-14

Advanced 5G Network Infrastructure for the Future Internet

Research and Innovation Action

Deliverable D7.3

Experimental Integration results of HW/SW modules of the overall SESAME framework

Document Number:	H2020-5GPPP-GA No.671596/*WP7/D7.3/30.09.2017*
Contractual Date of Delivery:	*30.09.2017*
Editor:	Antonino Albanese – ITL (Italtel S.p.A.)
Work-package:	*WP7*
Distribution / Type:	Public (PU) / Report (R)
Version:	1.0
Total Number of Pages:	63
File:	*SESAME_Deliverable 7.3_v1.0_Final*

Abstract

This document describes the experimental integration results of HW/SW modules of the overall SESAME framework.

The deliverable shows the functional blocks of the SESAME architecture that are mapped on HW and SW modules, as developed and tested during the activities carried out in WP3, WP4, WP5 and WP6.

Most of the PoCs are depicted and analysed deeply in other deliverables, used as references in the present document.

This deliverable will be used as "track" for the overall integration phase of the SESAME platform that will be depicted in the forthcoming Deliverable D7.4.

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	01.09.2017	Initial draft by ITL	A. Albanese
0.2	12.09.2017	Inclusion of contributions by ORION	E. Kafetzakis
0.3	13.09.2017	Inclusion of contributions by NCSRD	I. Giannoulakis
		Inclusion of contributions by i2CAT	P. Khodashenas
0.4	15.09.2017	Inclusion of contributions by UPC	Jordi Pérez-Romero
0.5	15.09.2017	Inclusion of contributions by ZHAW	Irena Trajkovska
0.6	18.09.2017	Inclusion of contributions by ITL	A. Albanese
0.7	20.09.2017	Inclusion of contributions by IPA	A. Whitehead
0.8	21.09.2017	Inclusion of contributions by ATOS	E. Jimeno
		Inclusion of contributions by CNET	L. Goratti
0.9	25.09.2017	Inclusion of contributions by FLE	H. Xiao
0.10	26.09.2017	General review by Editor	A. Albanese
0.11	26.09.2017	General full review	A. Dardamanis
1.0	27.09.2017	Full editorial and conceptual review by the project coordinator. Document ready for submission to the European Commission.	I. Chochliouros

Contributors

First Name	Last Name	Partner	Email
Antonino	Albanese	ITL	antonino.albanese@italtel.com
Claudio	Meani	ITL	claudio.meani@italtel.com
Pietro	Paglierani	ITL	pietro.paglierani@italtel.com
Marco	Beccari	ITL	marco.beccari@italtel.com
Emmanouil	Kafetzakis	ORION	mkafetz@orioninnovations.gr
Ioannis	Giannoulakis	NCSRD	giannoul@iit.demokritos.gr
Pouria	Sayyad Khodashenas	i2CAT	pouria.khodashenas@i2cat.net
Shuaib	Siddiqui	i2CAT	shuaib.siddiqui@i2cat.net
Athanasios	Chalas	i2CAT	athanasios.chalas@i2cat.net
Alfonso	Egio Artal	i2CAT	alfonso.egio@i2cat.net
Jordi	Pérez-Romero	UPC	jorperez@tsc.upc.edu
Juan	Sánchez-González	UPC	juansanchez@tsc.upc.edu
Oriol	Sallent	UPC	sallent@tsc.upc.edu
Ramon	Agustí	UPC	ramon@tsc.upc.edu
Irena	Trajkovska	ZHAW	traj@zhaw.ch
Alan	Whitehead	IPA	alan.whitehead@ipaccess.com
Elisa	Jimeno	ATOS	elisa.jimeno@atos.net
Leonardo	Goratti	CNET	lgoratti@fbk.eu
Cristina	Costa	CNET	ccosta@fbk.eu
Roberto	Riggio	CNET	rriggio@fbk.eu
Shah	Nawaz Khan	CNET	s.khan@fbk.eu
Tejas	Subramanya	CNET	t.subramanya@fbk.eu
Supreeth	Herle	CNET	s.herle@fbk.eu
Hui	Xiao	FLE	Hui.Xiao@uk.fujitsu.com
Athanassios	Dardamanis	SMNET	adardamanis@smart.net.gr
Ioannis	Chochliouros	OTE	ichochliouros@oteresearch.gr

Glossary

Acronym	Explanation
3GPP	Third Generation Partnership Project
4G	Fourth Generation of Mobile Communications
5G	Fifth Generation of Mobile Communications
AP	Access Point
API	Application Programming Interface
AR	Augmented Reality
ARM	Advanced RISC Machine
BF	Broadband Forum
BS	Base Station
CCTV	Closed Circuit television
CE	Crowded Event
CESC	Cloud Enabled Small Cell
CESCM	CESC Manager
CN	Core Network
CP	Control Plane
CPE	Customer Premises Equipment
CPU	Central Processing Unit
CQI	Channel Quality Indicator
C-RAN	Cloud-enabled RAN
CRUD	Create, Retrieve, Update, Delete
CWMP	CPE WAN Management Protocol
DB	Database, Data Base
DC	Data Centre
DL	Downlink
DP	Data Plane
DPI	Deep Packet Inspection
DSS	Decision Support System
E2E	End-to-End
EMS	Element Management System
EPC	Evolved Packet Core
ETSI	European Telecommunications Standards Institute
EuCNC	European Conference on Networks and Communications
FAP	Femto Access Point
FCAPS	Fault, Configuration, Accounting, Performance and Security
GA	Grant Agreement
GB	Giga Bytes
GbE	Gigabit Ethernet
GBR	Guaranteed Bit-Rate
GHz	Giga Hertz
GPRS	General Packet Radio Service
GPU	Graphics Processing Unit
GTP	GPRS Tunneling Protocol
GUI	Graphics User Interface
GW	Gateway
H2020	Horizon 2020
HO	Handover
HW	Hardware
HWA	Hardware Accelerator
I/O, i/o	Input/Output
ICT	Information and Communication Technology

ID, id	Identifier
IoT	Internet of Things
IP	Internet Protocol
ISO	International Organization for Standardization
IT	Information Technology
JSON	JavaScript Object Notation
KHz	Kilo Hertz
KPI	Key Performance Indicator
KVM	Kernel-*based* Virtual Machine
L1	Physical Layer
L2	Data Link Layer
LAN	Local Area Network
Light DC	Light Data Centre
LTE	Long Term Evolution
μs	micro-server
MA	Metric Aggregator
MAC	Medium Access Control
Mbps	Mega-bits per second
MEC	Mobile Edge Computing
MEC	Multi-Access Edge Computing
MHz	Mega Hertz
MOCN	Multi-Operator Core Network
NAS	Network Attached Storage
NFV	Network Functions Virtualization
NFVO	Network Functions Virtualization Operator
NMS	Network Management System
NNSF	Non-Access Stratum Node Selection Function
NOS	Network Orchestration System
NS	Network Service
NSD	Network Service Descriptor
NSNM	Network Service Notification Manager
OAI	Open Air Interface
OAM	Operations and Management
ODL	OpenDayLight
OPNFV	Open Platform for NFV
OS	Operating System
OVS	OpenvSwitch
PC	Personal Computer
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PHY	Physical Layer
PLMN	Public Land Mobile Network
PM	Performance Management
PNF	Physical Network Function
PoC	Proof of Concept
PoP	Point of Presence
PPP	Public-Private Partnership
Qemu, QEMU	Quick Emulator
QoS	Quality of Service
RAB	Radio Access Bearer
RAM	Random Access Memory
RAN	Radio Access Network
REST	Representational State Transfer
RDCP	Resource Data Collection and Processing
RIA	Research and Innovation Action

RISC	Reduced Instruction Set Computer
RLC	Radio Link Control
RRC	Radio Resource Control
RRM	Radio Resources Management
SATA	Serial Advanced Technology Attachment
SC	Small Cell
SC-C-VNF	Small Cell-Common-VNF
SCTP	Stream Control Transmission Protocol
SCVNO	Small Cell Virtual Network Operator
SD	Secure Digital
SDN	Software-Defined Networking
SDR	Software-Defined Radio
SFC	Service Function Chaining
SIM	Subscriber identity Module
SLA	Service Level Agreement
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SVM	Support Vector Machine
SW	Software
TCP	Transmission Control Protocol
TR	Technical Report
TS	Tenant Scheduler
UC	Use Case
UDP	User Datagram Protocol
UE	User Equipment
UL	Uplink
UP	User Plane
URL, url	Uniform Resource Locator
US	User Scheduler
USB	Universal Serial Bus
VA	Video Analytics
VBS	Virtual Base Station
vDPI	virtual Deep Packet Inspection
VIM	Virtualised Infrastructure Manager
VM	Virtual Machine
VNF	Virtual Network Function
VNO	Virtual Network Operator
VSCNO	Virtual Small Cell Network Operator
vTU	virtual Transcoding Unit
VTU	Video Transcoding Unit
VxLAN	Virtual Extensible LAN
WAN	Wide Area Network
WP	Work Package
WT	Warning Threshold
XML	Extensive Markup Language

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Deliverable outline

The present SESAME deliverable covers the Light DC prototype HW and SW integration, and includes the following sections:

- *Section 1* offers a brief introductory overview.
- *Section 2* recaps the overall architecture of SESAME, showing all the functional blocks of the system.
- *Section 3* describes the HW and SW modules developed in WP3, WP4, WP5 and WP6 that are integrated and tested for each one of the main blocks.
- *Section 4* provides some experimental results (e.g.: features, performance and capabilities) related to each one of the PoC and also testbed.
- Finally, *Section 5* summarises the "key topics" discussed in the document and concludes the deliverable.

1.2 Definitions of Terms and SESAME concepts

At this point, it is useful to provide definitions of terms and processes which will be used later in this document to describe the SESAME main concepts.

- **Small Cell (SC):** Does not change in the context of SESAME.
- **Execution infrastructure, micro-server:** Specific hardware that provides processing power, memory, storage and networking capabilities to the Small Cell.
- **CESC (Cloud Enabled Small Cell):** An intelligent entity composed of two, optionally co-located, network-connected physical devices: a Small Cell Physical Network Function (PNF) and a micro-server (μS) platform, which form a node in the distributed Light Data Centre (Light DC).
- **Cluster of CESCs:** A group of CESCs that are locally connected together, exchange information and are properly coordinated.
- **Light Data Centre (Light DC):** A micro-scale virtualised execution infrastructure that provides computational, networking and storage resources to the Small Cell.
- **VIM:** Manager of the HW and networking resources (lifecycle, provision, placement, operation) comprising a cluster of micro-servers, namely the Light DC, and the networking nodes and links (virtual and physical).

2 SESAME Architecture

The overall SESAME system architecture is depicted in Figure 1. The SESAME architecture foresees the functional split of the Small Cell in physical functions (SC PNF) and virtualised functions (SC VNF), based on the Multi-Operator Core Network (MOCN) requirements and associated Radio Resource Manager (RRM) and Operations and Management (OAM) features, which need to be supported. Further design decisions have leaded to the introduction of a new functional entity, named SC-Common VNF (SC-C-VNF). The SC-C-VNF is defined as a new element in the SESAME architecture that resides between the SC PNF and the different SC VNFs. This allows a unique SC-C-VNF per CESC, which performs control-plane multiplexing and coordination functions from the SC-PNF to the virtualised world. Each SC-VNF supports a single Virtual Small Cell Network Operator (VSCNO) and maintains its own control and user plane connections to the VSCNO's core network (CN). For a more detailed description, the reader is referred to [1].

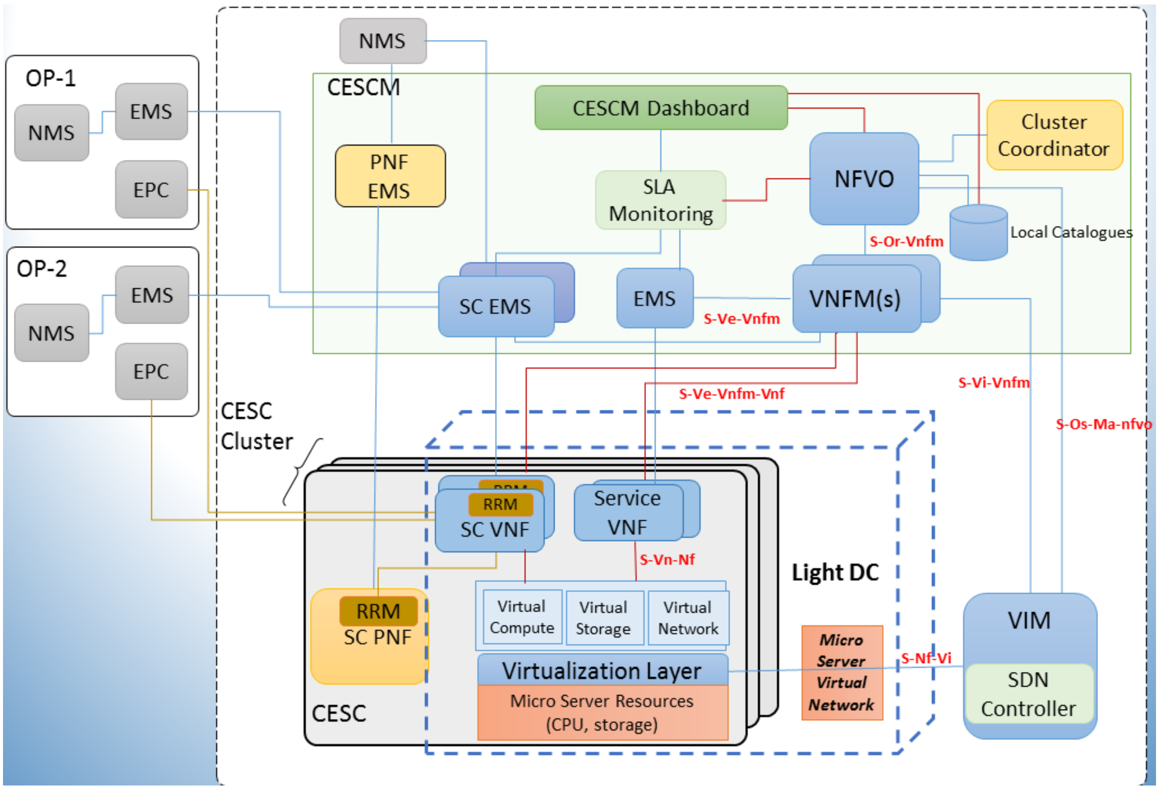

Figure 1: SESAME architecture

3 PoC description

The SESAME platform consists of a cluster of CESCs, which are devices that include both the processing power platform and the small cell element. Typically, CESCs are targeted for deployment at low- or medium-scale venues supporting multiple virtual network operators (i.e.: multi-tenancy). In addition, network services (NSs) and applications will be supported at the edge of the network based on the CESC processing capabilities. As depicted in Figure 1, the CESC offers the virtualized computing, storage and radio resources that can be made available for the virtual operators to provide local services. A cluster of CESCs is considered as a cloud from the upper layers. The assignment of cloud "slice" is the feature that enables multi-tenancy. The execution platforms are used to support VNFs that implement the different features of the small cells as well as to support Mobile Edge Computing (MEC) services.

In our scope, a CESC consists of 5G-*enabled* LTE-*based* femto base station (BS) with its standard backhaul interface, standard management connection (TR069[1] interface for remote management) and with necessary modifications to the data model (TR196[2] data model) to allow Multi-Operator Core Network (MOCN) radio resource sharing. The CESC is composed by a physical small cell unit attached to an execution platform based on one of x86[3], ARMv8[4] architectures, enhanced with Hardware Acceleration capabilities to support edge services that demand high performance capabilities. Edge cloud computing and networking are realized through the sharing of computation, storage and network resources of those micro-servers present in each CESC, which grouped together as-a-cluster form the Light DC.

This infrastructure provides also the backhaul and fronthaul resources for guaranteeing the requirements for connectivity in a multi-operator (multi-tenancy) scenarios. The hypervisor computing virtualization extensions enable access of virtual machines to the HW accelerators (HWAs) for providing an execution platform that can support the deployment of VNFs. The SESAME CESC management system is based on the ETSI NFV Architecture[5] enhanced for supporting multi-tenant operation. Different types of VNFs are envisaged, including multi-tenant Small Cell, for running the cognitive/self-x management operations to optimize CESC operation and for providing the mobile edge applications of the end-users. Each of these VNFs has its own

---

1 TR-069 (Technical Report 069) is a technical specification that defines an application layer protocol for remote management of end-user devices. It was published by the Broadband Forum (BF) and entitled CPE WAN Management Protocol (CWMP). More informative details can be found, *for example*, at: https://en.wikipedia.org/wiki/TR-069

2 TR-196 (Technical Report 196) is a BF technical specification. Its official title is "Femto Access Point Service Data Model." The purpose of this TR is to specify the Data Model for the Femto Access Point (FAP) for remote management purposes using the TR-069 CWMP.

3 x86 is a family of backward compatible instruction set architectures based on the Intel 8086 CPU and its Intel 8088 variant. More relevant information can be found, *inter-alia*, at: https://en.wikipedia.org/wiki/X86.

4 ARM is the industry's leading supplier of microprocessor technology, offering the widest range of microprocessor cores to address the performance, power and cost requirements for almost all application markets. Combining a vibrant ecosystem with over 1,000 partners delivering silicon, development tools and software, and with more than 90bn processors shipped, our technology is at the heart of a computing and connectivity revolution that is transforming the way people live and businesses operate. For more details also see: https://www.arm.com/products/processors.
The ARMv8 architecture introduces 64-bit support to the ARM architecture with a focus on power-efficient implementation while maintaining compatibility with existing 32-bit software. More related information can be found at: http://www.arm.com/products/processors/armv8-architecture.php

5 For more details also see: http://www.etsi.org/technologies-clusters/technologies/nfv

resource demands and KPI requirements that need to be coordinated by the CESCM. The combination of the proposed SESAME architecture provides the level of flexibility and scalability in the edge cloud infrastructure to meet these demands.

3.1 CESC

3.1.1 Light DC micro-server

The system proposed for the PoC provides the use of four different blocks acting as micro-servers, details can be found in [9]:

1. STM board[6] (please see Table 1) running a virtualization layer to host Common SC-VNF and SC-VNF (at least one).
2. Raspberry Pi 3[7] board.
3. NXP[8] LS2085A[9] board (please see Table 1) running a virtualization layer to host SC-VNF, Service VNFs (SW only) and storage.
4. INTEL node (Xeon v3[10]) equipped with a NVIDIA GPU M4000[11] for VTU HW acceleration and storage.

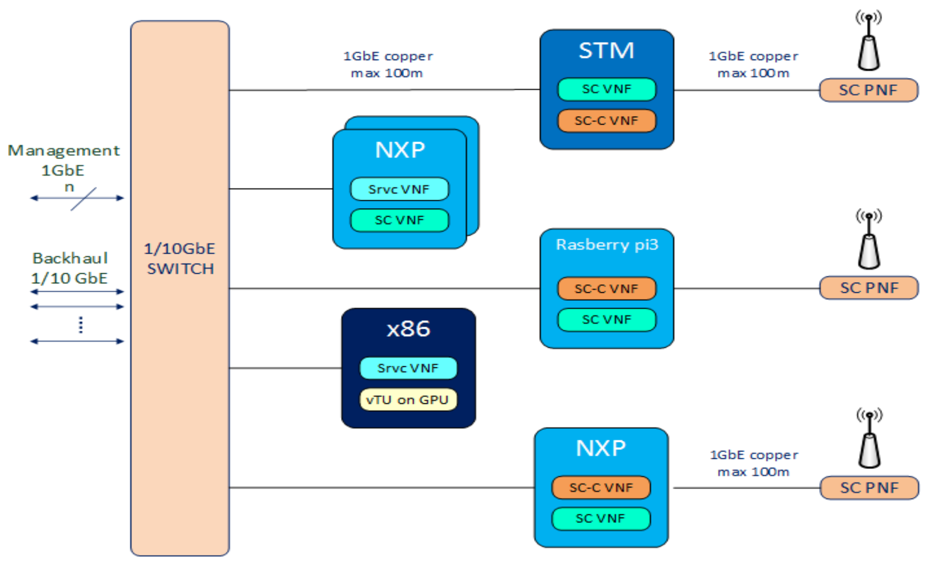

Figure 2: Light DC micro servers for the PoC

6 For more details see: http://www.st.com/en/evaluation-tools.html
7 For more details also see: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
8 More details about NXP can be found at: http://www.nxp.com/
9 More details about this specific NXP board can be found at: http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/qoriq-arm-processors/qoriq-ls2085a-rdb-reference-design-board:LS2085A-RDB
10 https://software.intel.com/en-us/articles/intel-xeon-processor-e5-2600-v3-product-family-technical-overview
11 See: http://www.pny.com/nvidia-quadro-m4000

Micro-server	Architecture	Cores	RAM	Storage	PCI-e[12] Acceleration	Clock Speed
NXP LS2085A	ARMv8[13], A57[14]	8	16 GB	500 GB	no	1.8 GHz
GOMA[15] (FlexPAC)	Xeon E5-2630v3[16]	8	64 GB	4+2 TB	GPU	2.4 GHz
STM board	ARMv8, A53[17]	4+1	1 GB	xx GB (SATA[18] disk)	no	1.3 GHz
Raspberry pi 3	ARMv8, A53	4	1 GB	xx GB (microSD[19])	no	1.2 GHz

Table 1: Micro-server - main characteristics

3.1.2 Small Cell VNFs

The Small Cell VNFs are key to the virtualisation of each VSCNO's network slice. They separate the control plane (CP) and user plane (UP) traffic, based on each VSCNO's PLMN ID, enabling each VSCNO's network slice to be isolated and managed separately. There is an SC VNF instance associated with each VSCNO and a single instance of the SC-Common VNF, which provides a coordinating role.

As described in Deliverable D2.3 ([2]), each SC VNF and SC-Common VNF lies on the S1 interface[20] between the SC PNF and the EPC of a particular VSCNO. From the SC PNF's perspective, the SC-Common VNF looks like an EPC; and from the Operator's EPC perspective the SC VNF appears as a standard single-operator Small Cell (eNodeB[21] or HeNB[22]). Each SC VNF may support multiple S1 connections towards the EPC (using S1-Flex). The SC VNF provides a full implementation of the Non-Access Stratum (NAS) Node Selection Function (NNSF) and is responsible for selecting an MME to serve each UE.

This architecture is illustrated in Figure 3.

[12] For more relevant information also see, *for example*: https://en.wikipedia.org/wiki/PCI_Express
[13] The ARMv8 architecture introduces 64-bit support to the ARM architecture with a focus on power-efficient implementation, while maintaining compatibility with existing 32-bit software. More related information can be found at: https://www.arm.com/products/processors/armv8-architecture.php
[14] For more details see: https://www.arm.com/products/processors/cortex-a/cortex-a57-processor.php
[15] For more details see: https://www.gomaelettronica.it/en/server-and-workstation-portables-high-density-storage-server-intel-i7-i5-i3-series
[16] http://ark.intel.com/products/83356/Intel-Xeon-Processor-E5-2630-v3-20M-Cache-2_40-GHz
[17] For more details see: https://www.arm.com/products/processors/cortex-a/cortex-a53-processor.php
[18] More related informative details can be found, *for example*, at: https://en.wikipedia.org/wiki/Serial_ATA
[19] Also see: https://en.wikipedia.org/wiki/Secure_Digital#Microhttps://en.wikipedia.org/wiki/Secure_Digital#Micro
[20] For further details see: http://lteguide.blogspot.gr/2011/11/s1-interface.html
[21] For more details see, *inter-alia*: https://en.wikipedia.org/wiki/EnodeB
[22] For more details see, *inter-alia*: https://en.wikipedia.org/wiki/Home_eNodeB

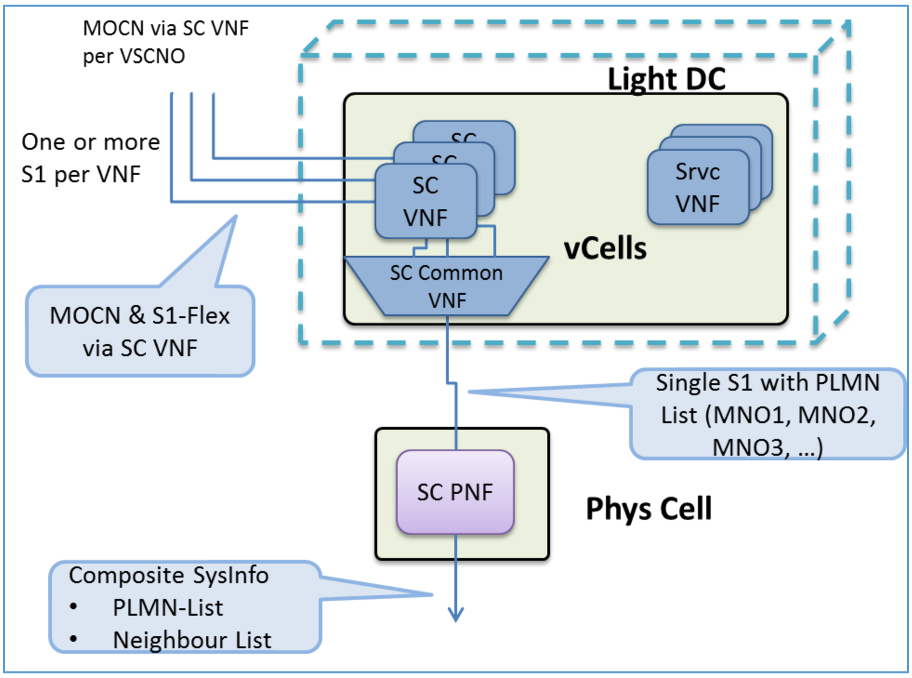

Figure 3: SESAME VNF architecture

3.1.2.1 SC-Common VNF

The SC-Common VNF is a "helper" function that supports the coordination of the SC VNFs.

Signalling

In the upstream direction (from the PNF towards the VNFs), the SC-Common de-multiplexes the S1 messages, directing them to the correct SC VNF based on PLMN-ID. In the downstream direction, the SC-Common simply merges all S1 messages into a common Stream Control Transmission Protocol (SCTP) connection.

RRM

The SC-Common monitors the RAB (Radio Access Bearer) assignments, modifications and releases for all tenants, for all causes (including handover) and maintains a resource view that is available to all SC-VNFs. Where appropriate, it optionally enforces PNF-*wide* limits on both the total number of UEs that may be served by the PNF and the total uplink and downlink user plane traffic.

For more details on the SC-Common VNF, please refer to Deliverable D2.3 (in [2]) and to the Deliverable D3.1 (in [3]).

3.1.2.2 SC VNF

All of the S1 signalling and user plane traffic for a particular VSCNO passes through the associated SC VNF. The main functions of the SC VNF are:

- It provides network isolation; the signalling and user plane traffic of the VSCNO are typically routed to a separate EPC.
- It provides load isolation; each SC VNF is hosted by a different virtual machine running on the Light DC such that, if one VSCNO is experiencing heavy load, this does not impact the performance of the SC VNFs of other VSCNOs.
- It implements the management of the VSCNO's network slice by implementing VSCNO specific UE and throughput caps.
- It provides the interface to the Service VNFs (see section 3.1.3); as all uplink (UL) and downlink (DL) user plane traffic passes through the SC VNF, it is able to optionally divert this traffic into a VSCNO specific service chain[23].

For more details on the SC VNF, please refer to Deliverable D2.3 (in [2]) and to the Deliverable D3.1 (in [3]).

3.1.2.3 Programmable RAN sharing Functionality

This section aims to provide the complete description of the development work and the hardware/software integration effort done for RAN sharing functionality. An initial description was already provided in [4] and [5].

RAN sharing is a key feature of the SESAME system that can maximise the benefits of the overall platform by hosting a multi-operator (multi-tenancy) environment. This environment is suitable to host small cell virtual network operators (SCVNOs) on top of the SESAME system. Two different ways were tackled by SESAME to develop a platform that can host multi-tenancy. The first is through the 3GPP MOCN feature. The second combines MOCN with resource slicing through RAN programmability. This approach allows to materialize the "Neutral Host" model in which multiple parties can be hosted in a single infrastructure made available by an infrastructure provider. The second option is explained hereinafter.

The complete RAN sharing function that was developed consists of the capability to accommodate in the programmable platform new tenants on the radio access network side. Therefore, not only different PLMN IDs are broadcasted to advertise to subscribers the multi-operator environment, but this is combined with the possibility to assign a slice of the available radio resources to each tenant. The assignment is done in a way to prevent potential conflicts among tenants, while accessing the resources. This task is not "trivial", but the resource blocks structure available at the radio access side allows segregation of the resources upon a tenant basis. Furthermore, radio resource slicing allows defining a different scheduler for each tenant, thus rendering the operators one another independent.

The experimental platform is built upon the open source software Open Air Interface[24] (OAI) small cell and 5G-EmPOWER ([16]) VIM, which are the principal elements of the advanced

[23] This is an important point. If desired, each VSCNO may implement a different service chain.

research prototype developed within SESAME to demonstrate radio resource slicing that can be achieved through RAN programmability. The additional components that were developed are also described, such as the template descriptor file that can be set as input to 5G-EmPOWER by the SESAME orchestrator.

Example of Descriptor files for 5G-EmPOWER VIM

5G-EmPOWER can support multi-tenancy in the form of slicing the radio resources. Users can use either the web-*based* dashboard or a REST[25] interface to perform CRUD[26] (Create, Retrieve, Update, Delete) operations on network slices. This section is used to show the slicing interface exposed by the 5G-EmPOWER Operating System (OS), and to explain the YAML[27] descriptors that can be used in order to instantiate new LTE network slices.

Figure 4 reports the resource manifest advertised by the 5G-EmPOWER OS for a network composed of one LTE eNB. Notice how the acronym VBS used in the manifest means "Virtual Base Station" and is the term used by 5G-EmPOWER to refer to one -or more- LTE eNB.

Each of the VBS advertised in this manifest comes with the following set of properties:

1. The physical address of the device (MAC address).
2. A human readable description of the device.
3. The geographical position of the device (latitude, longitude).
4. The list of cells available on the device together with their operating frequency and total number of physical resource blocks.
5. The number of physical resource blocks that are currently not allocated to any slice.

The resource manifest essentially advertises the resources that are available in the physical infrastructure. The slice owner can then use either the web-based management dashboard exposed by the 5G-EmPOWER OS or it can directly interface with the 5G-EmPOWER REST API. The latter option is particularly suitable in case the administrative commands are originating from a network service orchestration platform.

[24] Please refer to: *http://www.openairinterface.org/*
[25] Also consider the discussion within: *https://en.wikipedia.org/wiki/Representational_state_transfer*
[26] For further details also see, *among others,* the context proposed at: *https://en.wikipedia.org/wiki/Create,_read,_update_and_delete*
[27] Please see: *http://yaml.org/*

```
version: 1.0


VBSes:
 - name: VBS1
   description: human readable description
   hwaddr: 00:00:00:00:20:E1
   location:
     latitude: 46.0702531
     longitude: 11.1216386
   cells:
```


Figure 4: Template descriptor file for slice instantiation through 5G-EmPOWER

Demo use case: programmable RAN slicing for multi-tenancy

The RAN sharing demonstration scenario is shown in Figure 5. From the functional viewpoint, the demo allows creating two separate slices for downlink communications through RAN programmability. The reverse link (i.e., the uplink direction from the UEs to the small cell) was not changed since the most crucial part is indeed the DL transmission. The demo includes one small cell and two UEs connected. Two slices of the available radio resources are created through 5G-EmPOWER, hence allowing two different tenants over the same physical small cell. It is worth to remind that a tenant can be a SCVNO or any other entity offering services through the SESAME infrastructure. The amount of radio resources assigned to each tenant can be asymmetric and it does depend also on the overall system bandwidth of the system (i.e., up to 20 MHz in LTE). In other words, to each tenant can be assigned a different fraction of the overall resource blocks (i.e. slicing), as well as for a different amount of time. Hence, this creates a very dynamic environment to (re)configure tenant's resources.

The task of scheduling slices is assigned to a specific "Tenant Scheduler", which is a software entity created ad-hoc to serve this purpose. Within each slice, a different MAC scheduler can be instantiated for the sake of scheduling traffic addressed to different user terminals. In Figure 5, the Tenant Scheduler is denoted by TS, whereas the user scheduler by US. Each tenant is instead denoted as T1, T2, etc. Although the current development concentrated two slices, multiple can be instantiated adopting the same process. The process of creating slices consists of the 5G-EmPOWER VIM communicating with the 5G-EmPOWER eNB software agent that was specifically developed for RAN sharing and integrated with the small cell protocol stack. The VIM can create -or update- a slice through the commands exchanged with the software agent. For instance, the software agent, upon collecting performance statistics from the different layers of the protocol stack, can report them to 5G-EmPOWER, which can decide to increase the resources assigned to a tenant (provided that the SLA of other tenants is not violated, and all agreements are set in place for such a dynamic environment). After the resources have been assigned to a tenant, actual communications may begin.

As described above, specific YAML descriptor files were developed in order to enable a full automation of the RAN sharing functionality. As shown in Figure 5, relying on the descriptors in Figure 4, the SESAME orchestrator can optionally send to 5G-EmPOWER the slice descriptor, hence enabling tenants to automatically request for new resources.

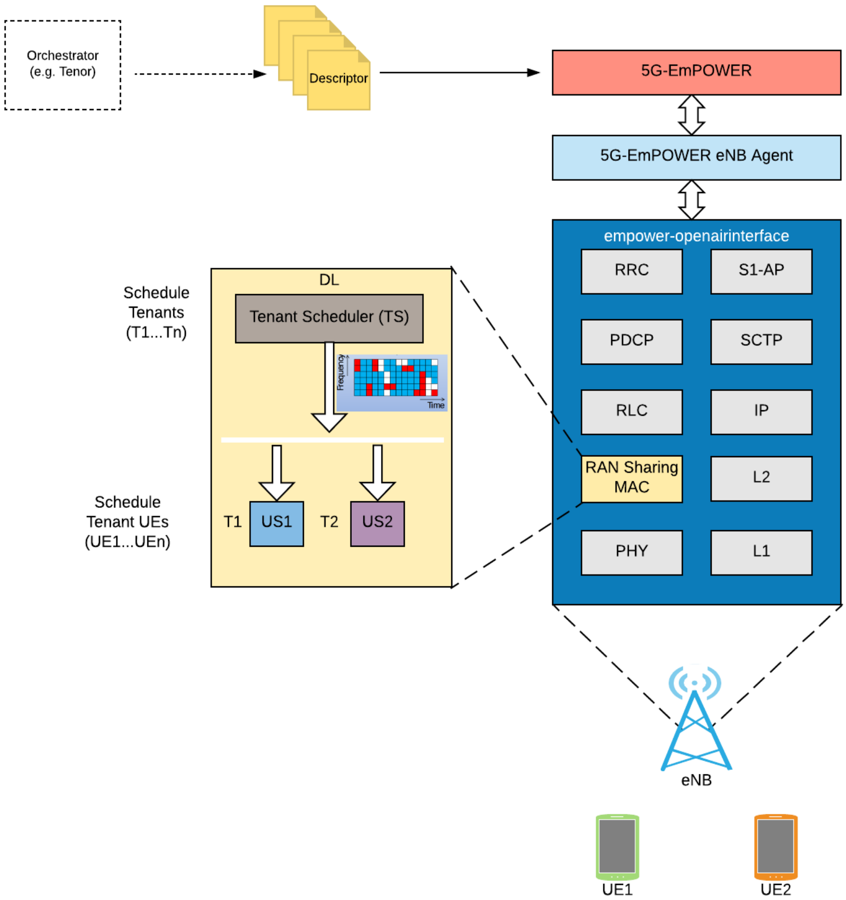

Figure 5: RAN sharing functionality implemented for demonstration

3.1.3 Service VNF

3.1.3.1 vDPI

The virtual Deep Packet Inspection[28] (vDPI) is designed to analyze in real-time network traffic, to recognize specific applications and to categorize each traffic flow according to its service. As this operation generates significant workload, it is highly recommended to be deployed in a medium to large VM flavor for efficient traffic processing. In the following, we describe the techniques that were employed.

For further details on the vDPI VNF, the reader is referred to [7].

[28] For more information about the more generalized concept of DPI (Deep Packet inspection) see, *for example*: https://en.wikipedia.org/wiki/Deep_packet_inspection

3.1.3.2 vWatermark

Watermarking deals with addition of (hardly perceivable) information to a digital input like text, images, videos or other. Most of watermarking techniques are designed to provide authentication or copyright protection for applications like rights management, media monitoring and distribution, and corporate or personal metadata association.

Usually only a few bytes are necessary to embed. Nevertheless, they should be inserted in secure manner, aiming for high levels of imperceptibility and robustness; the latter being understood as the difficulty of removing or altering the watermark without destroying it.

The vWatermarking VNF consists of two main parts, that is: a message being sent and the carrier media. Both of them carry important information for the consumer. A message can be described by three information roles: holding the provider (media producer), media and receiver (customer) specific data. These mostly consist of a unique identifier or an additional information of each role. The content of each message serves for controlling the proper use of the media (e.g. the producer's information for media origin identification, the receiver's for controlled distribution and the media information for guaranteeing the genuineness and integrity of the associated media data).

The testbed used for testing the vWatermark functionality is depicted in [13].

3.1.3.3 vTU

The Video Transcoding Unit (vTU) VNF provides some basic functionalities, which can be used to building video content sharing services for the users. In particular, the VTU provides video and audio transcoding functions, together with local storage capabilities of pre-recorded audio/video files.

The services provided by the vTU can be accessed through a web-service-*based* interface, available from any browser. Through the vTU, users can originate or receive live video streams. In particular, they can stream an originated video content to the vTU, or upload it as a pre-recorded file. Other users can receive this live content, or download it at a later time. If the originated video is shared among users in real-time, the VTU concurrently saves it to a distributed storage system, for successive content sharing. In the VTU, a common storage area is also dedicated to system managers that can upload contents to be offered to all the users. Such functionalities can be used to build enhanced video services, well-suited for the highly debated Crowded Event (CE) use cases. In this scenario, a high number of users concentrates in a small area for a short time, typically ranging from few hours to a week. Well-known examples of CEs are sporting matches or concerts at a stadium, congresses or exhibitions hosted by dedicated venues, international events spread over a university campus or even an entire city.

For further details on the vTU VNF, the reader is referred to [7].

3.1.3.4 vFirewall

A virtual firewall is a firewall service running in a virtualized environment, providing the usual packet filtering and monitoring services that a physical firewall would provide. Virtual firewall in

bridge-mode acts like its physical-world firewall “analog”. Positioned in a strategic point of the virtual network infrastructure, it can intercept virtual traffic destined for other segments.

For further details on the vFirewall VNF, the reader is referred to [7].

3.1.3.5 vVideoAnalytics

The two real-time video analytics-*based* VNFs (VA VNFs) have been designed and developed to investigate the two very import KPIs that 5G system aims to address, that is low end-to-end (E2E) latency and reduction in the backhaul consumption. These two real-time VA VNFs are intended for augmented reality (AR) services and video analytics-based real-time remote control of IoT devices (Smart IoT) services respectively, which are two examples of applications that require low end-to-end service latency and consume a large amount of real-time video streaming data to perform desired video analytics tasks.

For more detailed information regarding the descriptions and implementations of the two VA VNFs, the reader is referred to [8].

The information provided here focuses on the PoC description.

In the AR PoC, the AR VA VNF receives live video feeds from fixed CCTV cameras or mobile cameras mounted on smartphones, drones or robots.

The AR VA VNF offers an object-tracking-*based* AR service, (i.e. a predefined object of interest is tracked in the video frames). Upon the recognition of the object and the identification of its real-world location, the object of interest is tagged in the video frame and the information about the object, including its identity and the current location, is so added to the tagged area. After this kind of processing all the processed video frames are streamed to the relevant mobile devices that requested this AR service with unnoticeable latency between when the real-world video data is generated and when the processed video data is received by a mobile device.

The comparison between edge computing and remote cloud computing is demonstrated in terms of end-to-end service latency, i.e. when the AR VA VNF is deployed at the edge Light DC and when it is deployed on a third party’s remote cloud server. The comparison results showed that the remote cloud computing introduces longer end-to-end service delay, whereas the edge computing can ensure the AR video viewed on the mobile devices are in sync with that shown on the video generator.

This AR PoC was integrated with the SESAME testbed at NCSRD and demonstrated at EuCNC 2017[29].

3.1.3.6 vCache

This section aims to provide the complete description of the development and software/hardware integration work to develop the virtualised caching functionality. The initial description is already available in [4], [5] and [13]. Furthermore, an earlier version of the edge caching demonstration was already provided at the SESAME booth held at EuCNC 2017.

[29] For more details see: *http://eucnc.eu/?q=node/134*

Demo use case: content caching at the edge through light weight virtualisation

Multi-Access Edge Computing (MEC) offers a very interesting environment where to develop virtualised functions at the edge of the mobile network. From the perspective of MEC scenarios, it is worth to consider that the amount of computing and storage resources deployed at the edge of the network can vary significantly. In other words, some installations might concentrate significant amount of resources (i.e.: storage, computing and network), whereas other could be more limited. In any case, it could be a good common practice to be *as efficient as possible* in deploying virtualised network functions, especially for MEC. In this context, efficiency hinges around reduced consumption of CPU, memory storage and fast booting time. For this reason, the demonstration of virtualised caching relies on a lightweight virtualisation approach (if compared to standard virtual machines) based on Docker containers[30].

The functional view of the vCache functionality, including the necessary components introduced, is shown in Figure 6.

Kubernetes[31]: Orchestration of lightweight virtualised network functions

To develop the virtualised content caching demonstration, Kubernetes as the container orchestration tool was selected to develop the prototype of a CESC platform based on open source software. Kubernetes is itself an open source container management and orchestration platform developed by Google. It automates the deployment of container-*based* network services on top of a clustered infrastructure, scaling of those services to guarantee performance constraints, and taking care of service availability. Kubernetes uses a dedicated Master node for hosting the control and orchestration functions that manage the clustered infrastructure. The master node in turn controls the deployment of service VNFs (Docker containers) in the cluster. Kubernetes, within its managed clustered environment maintains an architecture where the collective resources of distributed physical machines are managed in a unified manner by the master node. Kubernetes uses a template file to define a network service (in YAML or JSON[32]) that specifies the deployment in sufficient details to enable the Kubernetes orchestrator to instantiate containers in the desired numbers, location and constraints.

From the perspective of managing the virtualised functions, the implementation of the vCache functionality is composed of a single Master node and three Worker nodes. Each worker node consists of a POD[33] with a single container in them (A POD is a group of one or more containers, with shared storage and network, with a specification on how to run the containers). One of the machine acts as both the Master node and the Worker node, whereas the other two are just the Worker nodes.

[30] For further details see: *https://www.docker.com/what-docker*
[31] Please see: *https://kubernetes.io/*
[32] Please see: *http://json.org/*
[33] For more details also see: *https://kubernetes.io/docs/concepts/workloads/pods/pod/*

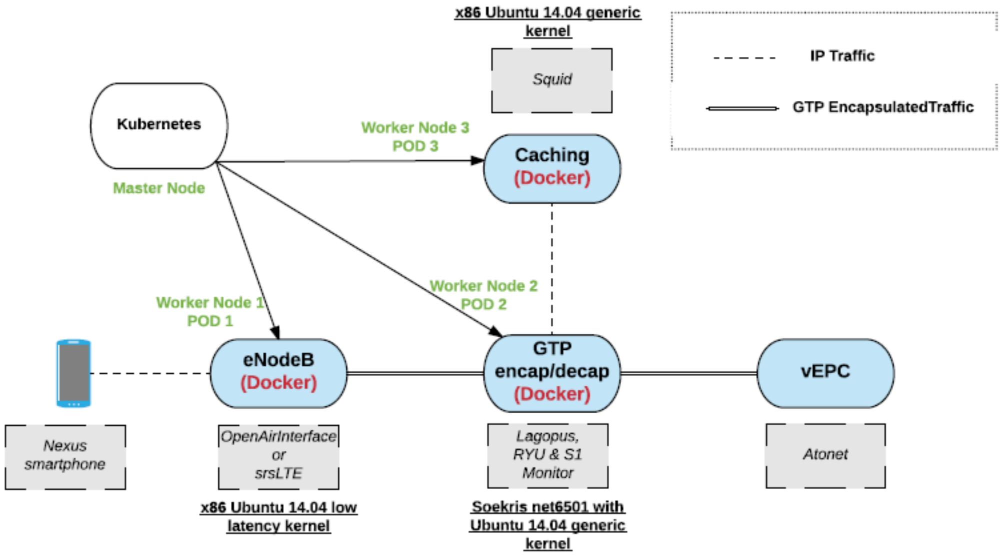

Figure 6: Functional view of the vCache demonstration components

3.1.4 Small Cell PNF

There is a single SC PNF connected to each CESC. In the PoC, the Small Cell PNF is provided by the ip.access E40 LTE Access Point (as in [15]).

3.2 CESCM

3.2.1 CESCM Portal

The CESCM portal is the front-end manager of the SESAME services and underlying infrastructure. SESAME, being a project oriented to the Mobile Edge Technology, combines a variety of technologies, such as NFV and network management. The portal is based upon the functionality of the CESC core components, namely the NFV orchestrator, the monitoring and the EMS of physical resources IPA's NOS (Network Orchestration System).

3.2.2 VIM

For the VIM PoC description please refer to [10].

3.2.3 SDN Controller

The role of the SDN controller in the PoC scenario is to provide a full control of the VIM infrastructure consisting of OpenStack[34] physical nodes (control, compute and neutron) and a physical SDN-*enabled* switch that also hosts Netfloc[35] (SDN controller).

[34] For further details see: https://www.openstack.org/
[35] See: http://icclab.github.io/netfloc/

Apart from allowing full network connectivity and integration in OpenStack VIM, Netfloc contains a library in order to support traffic steering that is based on the principle of port matching and MAC-to-virtual addresses rewriting in order to provide end-to-end SFC support in OpenStack VIM. The SFC library and functionality has been already validated in a wide SFC-WAN scenario within the scope of T-NOVA[36] as a reference project, for the support for intra-datacenter chaining of multiple VNFs like: traffic classifier transcoding unit, watermarking, virtual proxy service, etc. Within SESAME, an SFC demo was shown in the EuCNC 2016 demonstration booth[37] in cooperation with the COHERENT 5G-PPP project[38] (see Figure 7).

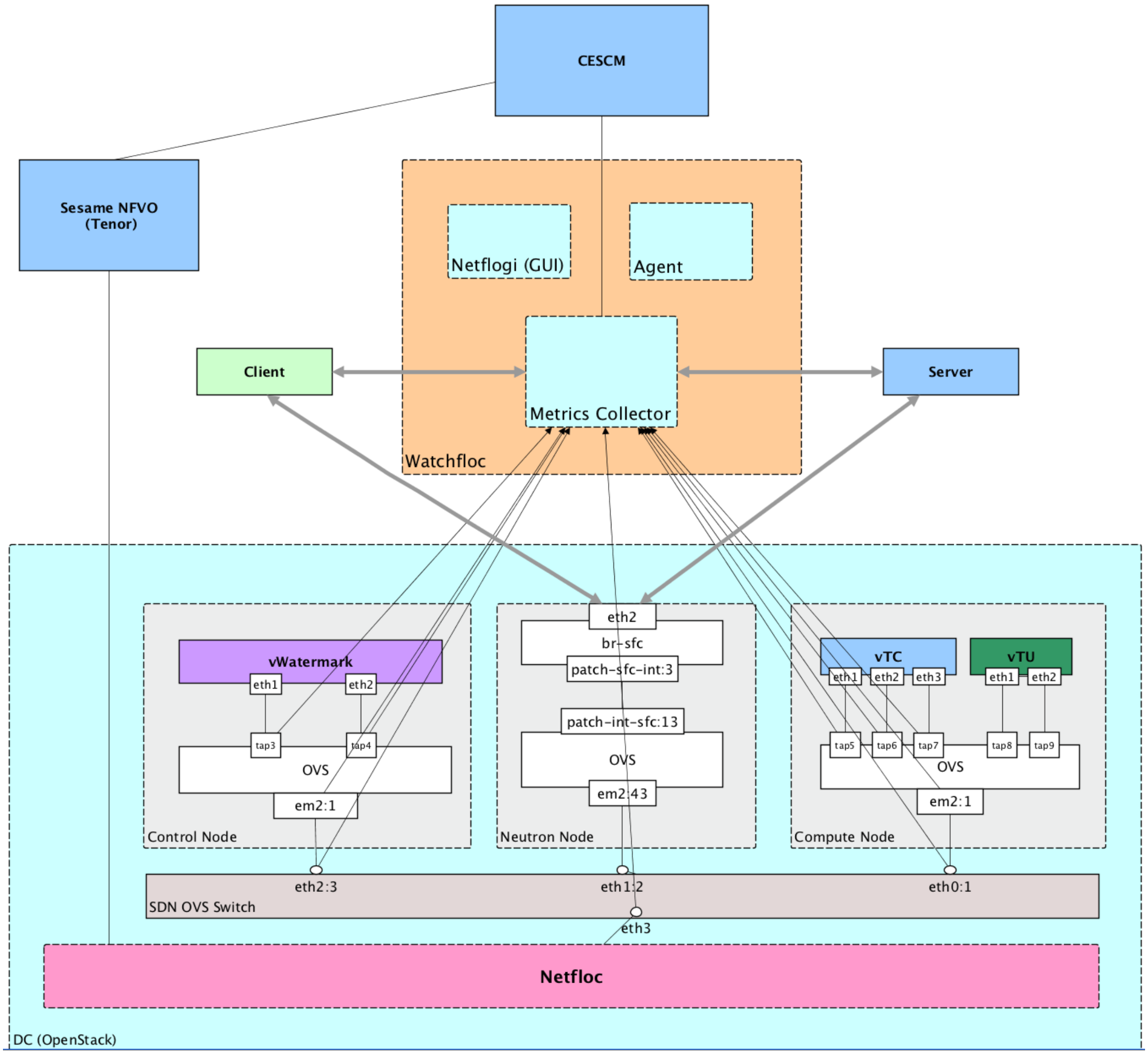

Figure 7: SESAME Service Function Chaining demo use-case setup

In this part, it is described one of the use-case scenarios that is associated to the cloud part and the VIM. The objective in this demo is to show the service function chaining, involving SESAME-

36 For more details about the T-NOVA project see: http://www.t-nova.eu/
37 More details can be found at: http://www.eucnc.eu/2016/www.eucnc.eu/indexe637.html?q=node/134
38 For more details about the COHERENT project see: http://www.ict-coherent.eu/

made VNFs and Netfloc's SFC algorithm in action. It also has the objective to show relevant metrics associated to the service and the network topology, which can serve as sanity indicator for CESCM. CESCM can configure alerts given certain thresholds and in function of this, trigger TeNOR[39], the SESAME NFVO, to perform actions in accordance to the SLAs.

3.2.4 NFV Orchestrator

The multi-tenant mixed radio-cloud environment of SESAME poses extra challenges for service management and orchestration, especially on ensuring the Quality of Service (QoS) per tenant. The logical C-RAN manager/orchestrator needs to simultaneously take into account both, the radio status (i.e.: volume of traffic, geographical distribution of traffic, etc.) and the cloud capabilities (i.e.: available IT resources, VM to VM communication requirements, etc.) for all the action related to the service lifecycle management. To do so, forming a feedback QoS ensuring loop is necessary. Such a loop consists of three main steps:

- Monitoring: A phase in which performance monitoring parameters are collected from the radio/cloud elements (e.g. SC physical network function, virtual machines, etc.) and handed over to the decision-making process. Depending on the nature of the resources, (i.e. radio or cloud, QoS requirements, the available Service Level Agreement (SLA), etc.) the monitoring parameters might vary from one use case to another.
- Decision-making: A phase in which performance metrics collected in the previous step are processed. Depending on the situation and available resources, a decision will be taken to ensure the level of QoS (with the help of a dedicated algorithm). Besides available resources, in a multi-tenant scenario the decision-making process needs to take into account the status of other tenants (i.e. keeping a good level of QoS for one tenant should not cost the failure of others). In principle, the nature of such a decision-making algorithm can range from a greedy heuristic to a complex cognitive form.
- Reaction: Upon making a decision, the management/orchestration system needs to coordinate the interaction with the other lower level modules such as Element Management System (EMS), Virtual Network Function Manager (VNFM) and Virtual Infrastructure Manager (VIM) to react appropriately.

Ensuring QoS in the SESAME multi-tenant cloud-*enabled* RAN is the main focus of WP5 and WP6. The proposed solution to ensure the QoS in the joint radio cloud environment of SESAME is described as a pseudo-algorithm, where we proposed a performance evaluation of edge cloud services. The evaluation mechanisms are based on two types of actions:

- Preventive (proactive) action: foreseeing saturation levels that can potentially lead to QoS breaches, warning alerts are sent to the appropriate management module for it to consider performing corrective actions e.g. VNF scaling, and;
- Correction (reactive) action: in case of breach in the QoS, violation alerts are reported to be analysed and corrected, if possible. Figure 8 depicts the SESAME proposed QoS insurance feedback solution.

Network Services in SESAME are composed of a SC-PNF, a SC-VNF and a chain of service VNFs, which are described in the NSD (Network Service Descriptor). Therefore, in the monitoring phase, the proposed solution (Figure 8), will retrieve metrics from both cloud and radio

[39] More details about TeNOR can be found at: https://github.com/T-NOVA/TeNOR/wiki

parameters. The Metric Aggregator (MA), as its name indicates, is the function responsible for combining and filtering the collected monitored parameters and associating them with the running services over the SESAME platform. The Metric Aggregator continuously processes the collected monitoring values for the QoS or SLA evaluation. Depending on the use case, the algorithm used for this purpose might be a simple threshold checking logic or a complex multi attribute decision-making process. Bearing in mind its fast processing time, we select the threshold checking procedure. To this end, a threshold, i.e. Warning Threshold (WT), is defined for each of the monitored metrics, aiming to detect the critical values per case and trigger a warning alert for the specific failure component. Note that, WTs are defined in a way that the multi-tenant capabilities of the system are taken into account. In this sense, the proposed solution is inflexible, i.e. values of WTs will not adapt dynamically according to the real-time needs of the system. Having said that, SESAME targeted the introduction of a more complete solution.

In conjunction with a more complex data process, SESAME envisioned a module denoted as the Decision Support System (DSS), as shown in Figure 8. The main responsibility of DSS is to detect the level of severity on the QoS evaluation process done in MA and decide whether a reactive or a proactive action is needed. Basically, such a decision will be made based on the high level SLA agreements made with VNOs. Such an agreement will indicate points such as at what PoP a VNO will be present, how much of overall IT resources are dedicated to a VNO, etc. With the help of DSS, the SESAME solution, in addition to the "per NS performance metrics" will bring in the high level SLA agreements into calculations.

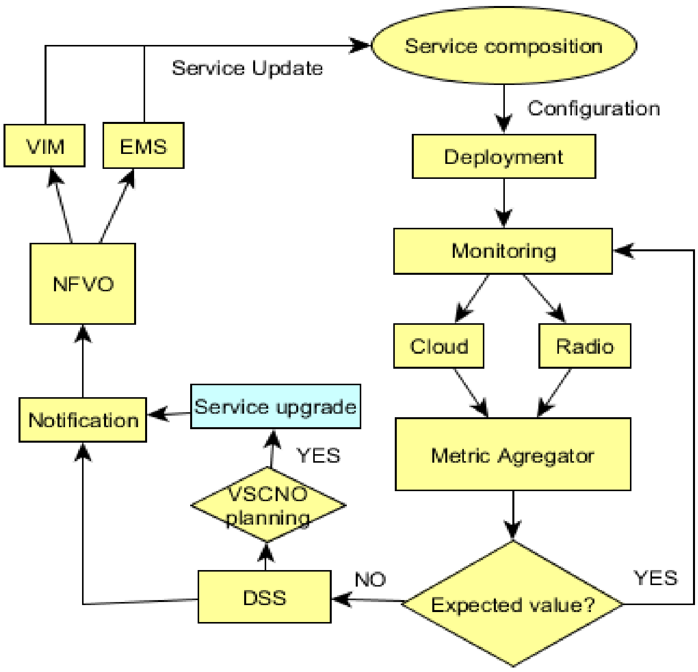

Figure 8: SESAME QoS insurance feedback loop

NFVO is the responsible module to perform the appropriate reaction based on the analysis made in the previous steps. There are a set of possible reaction mechanisms, including:

- Reconfiguring the flow of data in a NS (i.e. changing the SDN rules);

- migrating the NS within the point of presence (PoP) of from one PoP to another;
- scaling up/down the whole NS (i.e. instantiation of a parallel service or terminating a running one);
- scale in/out of VNF (i.e. adding more resources to a VNF).

Depending on the nature of requested process (proactive/reactive), nature of warning (radio/cloud), available resources, possibilities of VNFs indicated in the VNFD by developers, NS agreement with VNO depicted on NSD, NFVO selects and applies a reaction mechanism. Figure 8 depicts also the workflow of the reaction process. That is, NFVO has the possibility to interact with the EMS (managing the radio parameters of SC or SCs) and/or VIM (managing the cloud infrastructure – e.g., OpenStack). In this sense, EMS and VIM are the enablers' components for the adaptation service of NFVO on cloud and/or radio domain. It is worth noting that, in a more advance scenario, EMS and VIM can also be enhanced with self-x features, aiming to enable them to make local decisions for a CESC or a set of CESCs (sub set of Light DC).

From the infrastructure owner perspective, besides the status of each provided service, the overall system performance is also highly relevant; therefore, MA is also able to expose general information about the system status. This information helps the infrastructure owner to perform a temporary capacity upgrade if the expected use of resources does not meet planned terms, e.g. due to overloaded use of services / users. The data can be seen visually from the CESCM Portal through for example a customized dashboard.

The preliminary result of the integrated SESAME QoS solution has been presented on the SESAME EuCNC 2017 demo. In addition, based on the SESAME consortium decisions new showcases will be presented on the SESAME final demo. More details of this integration has been presented in [12] and, with less details, will be presented in the following sections.

3.2.5 EMS

The EMS is the entity in charge of the key functionalities as fault, configuration, accounting, performance and security (FCAPS). It manages the traffic between the different network elements, coordinating configuration of multiple devices. The EMS associated to radio functions also includes autonomous self-x functionalities to reconfigure the mobile network.
The SESAME EMS is based on the ip.access Network Orchestration System (NOS), which provides the functions of both the PNF EMS and SC EMS. The NOS is a client – server solution that comprises a single logical server that connects to each managed element and one -or more-client instances that provide a graphical user interface to users; for all details, please refer to Deliverable D5.2 (as in [9]).

3.2.5.1 Knowledge Discovery capabilities in the EMS

The introduction of knowledge discovery capabilities in a wireless network provides the ability to smartly process input data from the environment and come up with knowledge that can be formalized in terms of models and/or structured metrics that represent the network behavior. This allows gaining in-depth and detailed knowledge about the network, understanding hidden patterns, data structures and relationships, and using them for a making smart network planning and optimization decisions. In this way, the extracted knowledge models can be used to drive the decision-making of the actions associated to different self-x functionalities.

Knowledge discovery is supported by machine learning tools to perform the mining of the data. Extracted knowledge models can be defined at different levels: cell level (contains the characterization of the conditions on a per cell basis); cell cluster level (characterization of groups of cells built according to their similarities), and; user level (contains the characterization of the conditions experienced by individual users).

Based on the above, Figure 9 presents the considered framework for demonstrating the introduction of knowledge discovery capabilities in the context of SESAME. It is associated to the EMS, which encompasses both the PNF EMS and the SC EMS modules of the SESAME architecture.

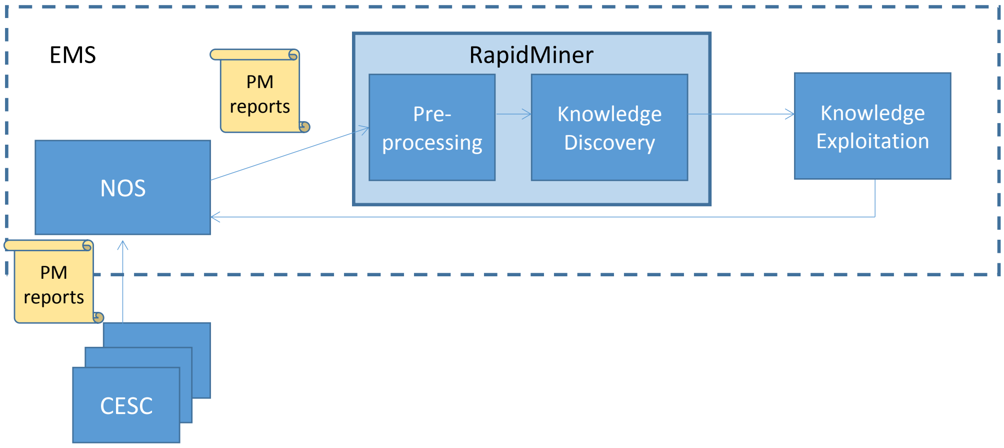

Figure 9: Demonstration of knowledge discovery in SESAME

3.2.5.1.1 Performance Management (PM) reports

Performance Management reports are XML files conforming the format described in [20]. They are produced according to a configured Reporting Interval and each file contains one or more Granularity Periods. The Granularity Period defines the time frame across which measurements are collected and aggregated and is typically defined to be in the range of 5 minutes to 24 hours. The granularity period of all the sample data in this study was set to one hour. The Reporting Interval is typically a multiple of the Granularity period. In the sample data used for this study, the Reporting Interval was set to either one hour or 24 hours.

Within each file, performance measurements are organised into groups of related items known as packages. Each file may contain up to 128 performance counters organised in to 27 packages. Some example packages are: (i) Access Control and Admission Control Packages that record the number of attempted and failed attempt to access the cell and establish radio bearers; (ii) Hand-in and Hand-out packages that record the number of attempted and successful handover procedures; (iii) A GTP-U[40] Usage Package that records the number of uplink and downlink GTP-U packets sent and received, the number of packets lost plus the total number of octets sent and received; (iv) A User Plane Package that records the number of call attempts by Radio Access Bearer (RAB) type, the maximum and mean number of simultaneous calls, uplink and downlink bandwidth utilisation by RAB type.

[40] For further details also see: http://lteworld.org/specification/gprs-tunnelling-protocol-user-plane-gtp-u

3.2.5.1.2 Pre-processing, knowledge discovery and knowledge exploitation

The pre-processing stage takes as input the PM files generated by the NOS and extracts the relevant metrics to be used by the knowledge discovery depending on the use case in hand. For that purpose, this stage can combine multiple PM files associated to different cells and/or time periods. Then, the knowledge discovery stage includes the machine learning algorithms to carry out the mining of the input data and extract the knowledge models.

Both the pre-processing and the knowledge discovery stages are implemented by means of the RapidMiner Studio Basic tool [21]**.** It is a powerful visual design environment for rapidly building complete predictive analytic workflows and incorporates multiple pre-defined data preparation and machine learning algorithms.

Finally, the knowledge exploitation stage applies the obtained knowledge models to drive the decision-making associated to different self-x functionalities. As shown in Figure 9, this stage can interact with the NOS to configure specific SC parameters.

3.2.5.1.3 Demonstration Use case: Energy saving

The considered use case to illustrate the operation of the proposed framework is the energy saving self-x functionality, which intends to reduce the overall energy consumption associated to the small cells deployed by the SCNO. In this case, the energy reduction is achieved by switching off the cells that carry very little traffic at certain periods of the day (e.g. at night) and making the necessary adjustments in the neighbour cells so that the existing traffic can be served through another cell. In this context, the knowledge discovery framework applies a classification methodology for identifying candidate cells to be switched-off based on their time domain traffic patterns. The automation of this procedure based on expert criteria captured in a training set becomes particularly useful considering that networks in the envisaged ultra-dense scenarios for future 5G systems can comprise several tens of thousands of cells. Therefore, it is not practical that a human expert can make this classification manually.

Based on the above, the classification categorizes the cells in the following classes:

- Class A: Candidate cell to be switched off.
- Class B: Cell that cannot be switched off.

It is worth mentioning that the final decision on whether or not to switch off a cell will make use of this classification as well as other possible inputs which are out of the scope of this work (e.g. the neighbour cell lists to ensure that traffic generated in a cell that has been switched-off can be served through another cell).

In the following, the different steps of the proposed classification approach are described. For the specific mathematical details the reader is referred to [22].

a) Pre-processing stage

The PM files generated by the NOS system include a number of XML files corresponding to different cells and periods of time. Each XML file includes metrics associated to different time instants. In turn, the considered classification process is based on the time domain traffic pattern of the different cells. For that purpose, the pre-processing stage is responsible for extracting the relevant metric to be used in the classification and presenting them in a format that is understandable by the classifier. Specifically, the selected metric considered in this work is the number of RAB admissions that have been accepted in a cell. Then, the pre-processing

stage builds, for each cell, a time series $\mathbf{X}_i=(x_i(t), x_i(t\text{-}1), ..., x_i(t\text{-}(M\text{-}1)))$ composed of M samples of the number of RAB admissions in the cell i at different times t with a certain granularity.

Figure 10 illustrates the building blocks of the pre-processing stage implementation using RapidMiner. The first block (Loop XML Files) reads each of the input XML files, while the subsequent blocks perform different operations to merge multiple XML files, to select from each one the hourly samples of the number of RAB admissions, and to build the table with the pre-processed data.

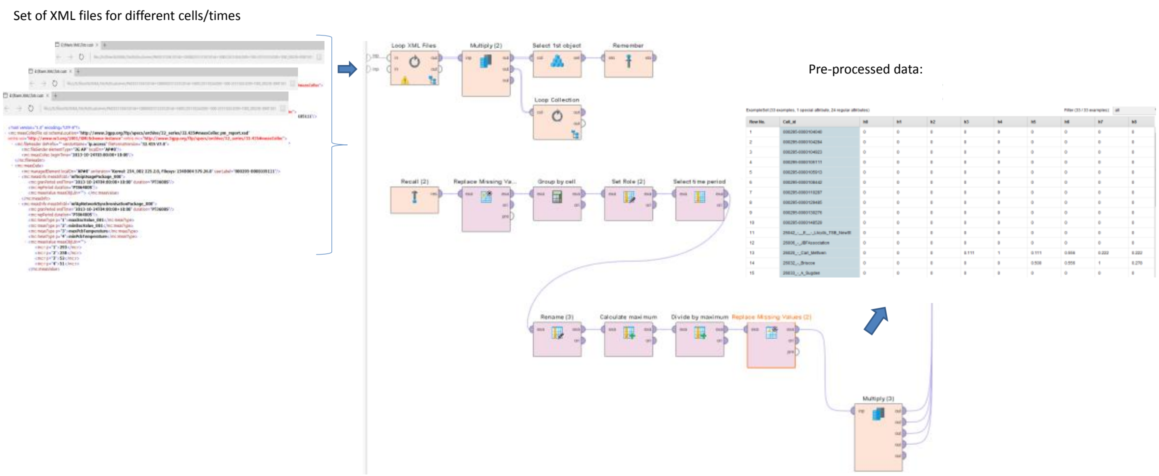

Figure 10: Pre-processing stage

b) Classification stage

The classification stage performs the association between the input time series $\mathbf{X}_i$ of the i-th cell and the class $C(\mathbf{X}_i)\in\{A,B\}$ of the cell. The internal structure of the classifier is given by the specific classification tool being used and its settings are automatically configured through a supervised learning process executed during an initial training stage. This training uses as input a training set composed by S time series $\mathbf{X}_j$ j=1,...,S of some cells whose associated classes $C(\mathbf{X}_j)$ are pre-defined by an expert. The supervised learning process will analyse this training set to determine the appropriate configuration of the classification tool. In this way, the resulting classifier after the training stage can be used for classifying other cells whose class is unknown.

Figure 11 illustrates the RapidMiner modules implemented for the performing the classification stage. The first module is the training stage that reads the cells from the training set and injects them to each classification in order to build the classification model (i.e., this is done in the first module shown inside a classification algorithm). Then, the last module of each classification algorithm takes as input the small cells to be classified from the pre-processing stage and applies the obtained classification model.

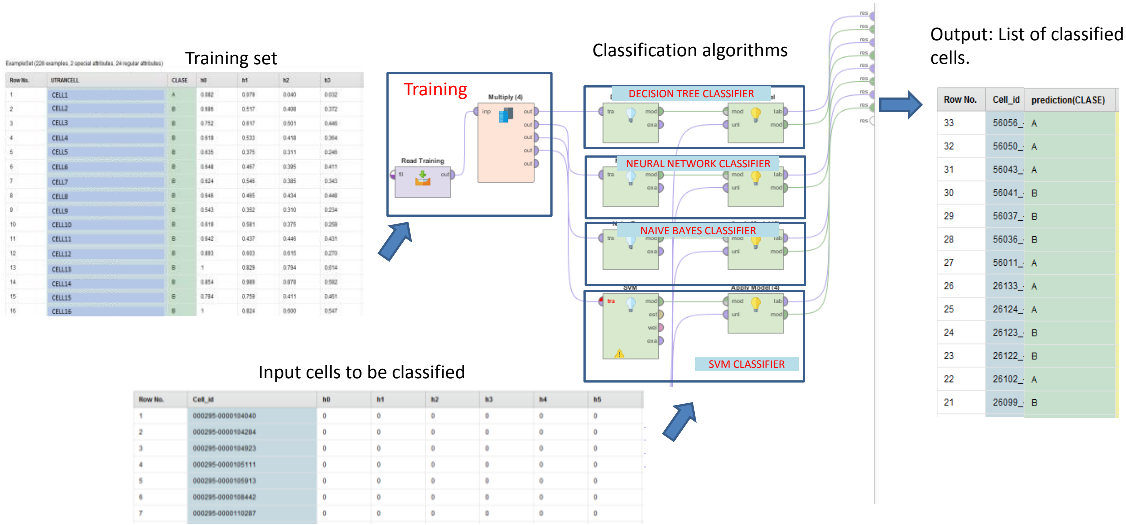

Figure 11: Classification stage

As shown in Figure 11, the following classifiers are implemented [23]:

- Decision tree induction: The classification is done by means of a decision tree, which is a flow-chart structure where each node denotes a test on a feature value, i.e. a component of vector $\mathbf{X}_i$, each branch represents an outcome of the test, and tree leaves represent the classes. The tree structure is built during the supervised learning stage through a top-down recursive divide-and-conquer manner.
- Naive Bayes classifier: The classifier evaluates the probability Prob(C($\mathbf{X}_i$) | $\mathbf{X}_i$) that a given cell $\mathbf{X}_i$ belongs to a class C($\mathbf{X}_i$) based on the values of the components of $\mathbf{X}_i$. The resulting class is the one with the highest probability. The computation of this probability is done using Bayes' theorem[41] under the assumption of class conditional independence. The different terms in the computation of the Bayes' theorem are obtained from the analysis of the training set.
- Support Vector Machine (SVM): A SVM is a classification algorithm based on obtaining, during the training stage, the optimal boundary that separates the vectors $\mathbf{X}_j$ of the training set in their corresponding classes C($\mathbf{X}_j$). This boundary is used to perform the classification of any other input vector $\mathbf{X}_i$. The optimal boundary is found by means of a nonlinear mapping to transform the original training data into a higher dimension so that the optimal boundary becomes a hyperplane.
- Neural Network: The classification is done by means of a feed-forward neural network that consists of an input layer, one or more hidden layers and an output layer. Each layer is made up of processing units called neurons. The inputs to the classifier, i.e. each of the components of vector $\mathbf{X}_i$, are fed simultaneously into the neurons making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer. The process is repeated until reaching the output layer, whose neurons provide the selected class C($\mathbf{X}_i$). The weights of the connections between neurons are learnt during the training phase using a back propagation algorithm.

[41] More details can be found, *inter-alia* at: https://en.wikipedia.org/wiki/Bayes%27_theorem

3.2.6 SLA monitoring

The evaluation of SLA dictates the quality and type of service expected by the consumer, provided from the infrastructure. The SLA Framework provides tools and best practice guidance on the SLA lifecycle for multi-clouds operators. It helps to materialize the contract between contractors by providing:

- Standardized SLAs Definition.
- Multi-Cloud Independent Monitoring.
- SLA Template Definition.
- Multi-Cloud SLA Assessment, Enforcement Accounting and Notifications.

All these features provide the baseline reference aligned with ISO to start with for the legal contract, the definition of metrics and a technology framework capable of managing the SLA lifecycle providing in Multi-Cloud Scenarios.

The SLA evaluation is composed of several modules inside the CESCM; in fact, Figure 12 depicts the internal infrastructure of the components. The communication between modules has been designed to provide a management workflow between all the components and specially infrastructure management component of the platform, in order to assure the best quality of the service in a lower reaction time, when some resources of the infrastructure is not working as expected. This communication is performed through internal components acting as APIs REST.

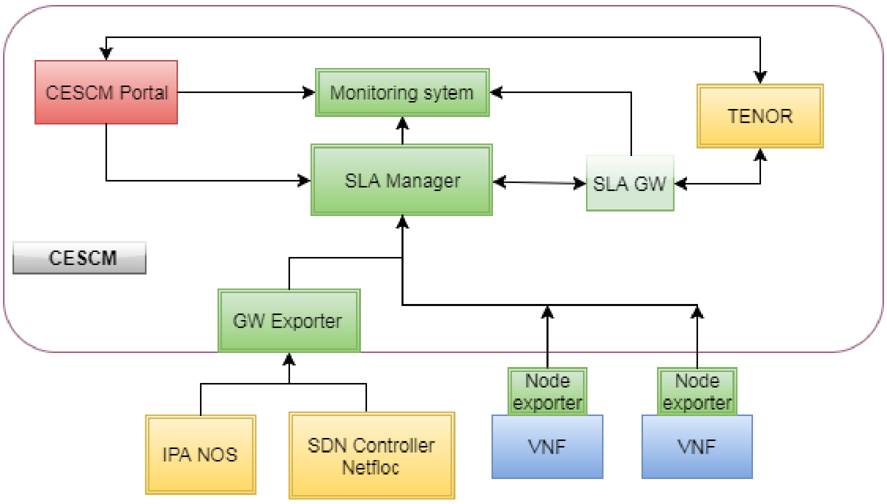

Figure 12: SLA components

The user is able to contract the available services in the platform from the CESCM Portal, where different services and tiers are listed and contracted. The negotiation is performed between the user and service provider. The SLA Manager makes use of external monitoring system that is able to monitor different infrastructure and components.

Cloud SLA Monitoring gathers KPIs from the Service VNFs forming a Chain. The individual information of each VNF is combined to create service KPIs; this is then compared with the agreed SLA. If the system encounters some problems in the monitored parameters previous

mentioned, then it sends a notification to the orchestrator to proceed with the needed action; if the problem persists the system notifies the user about the violation produced in the CESCM Portal.

The same situation is encountered when an alarm is raised from the other monitored infrastructure. A Gateway exporter has been developed in order to act as translator of the SDN controller module and the Element Management System of the small cell cluster. The functioning of the small cell is monitored through a tailored service for the SESAME platform; parameters are evaluated in the smart monitoring system so as if the configuration of the EMS does not comply with the expected values, it is possible to reconfigure in order to provide the QoS expected by the service provided. The network service is monitored in though the Netfloc exporter.

4 PoC experimental results

SESAME pilot will demonstrate a scenario as presented in Figure 13. Using the SC Common VNF and SC VNF, it is possible to intercept S1 bearers of VSCNOs both on the up- and down-links. More specifically, SESAME will provide enough means to de-/en- capsulate data packets from/to the S1 radio bearer, which allows having edge services on the Light DC on per VSCNO basis.

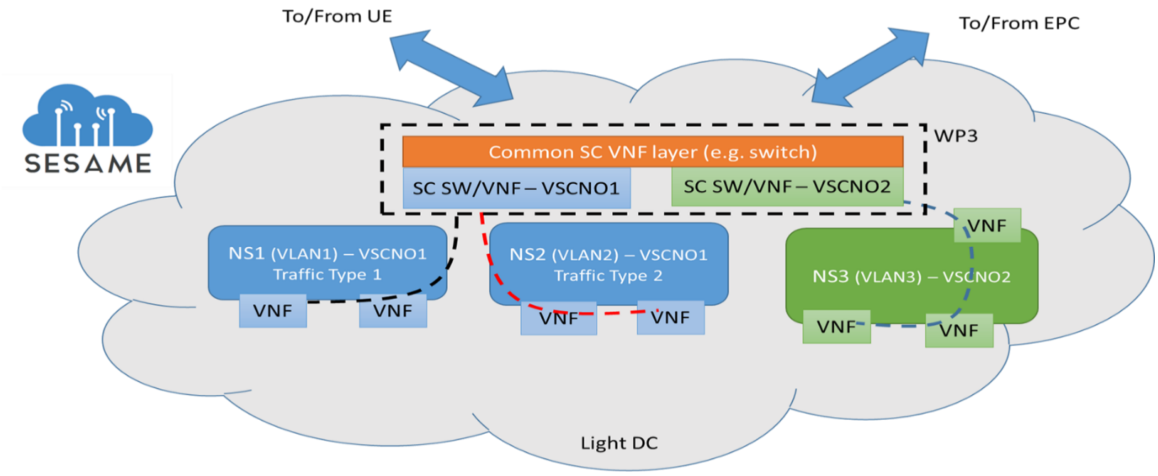

Figure 13: Network Services

SC VNF is the only exchange point where the radio traffic of a VSCNO enters/departs to/from the cloud world. Therefore, all NSs of a VSCNO should include it necessarily (one SC VNF per VSCNO is assumed to be enough). As presented in Figure 4, where dash lines stand for the data flows, this point is reflected on the SESAME NSD and then, via the NFVO and VIM, should be enforced at each service instantiation event. SESAME PoC will show how this situation will be handled in a multi-tenant scenario, e.g. for three (3) VSCNOs each having its own set of edge services running on the Light DC.

In addition to multi tenancy, SESAME PoC aims to illustrate the establishment of the complete chain of monitoring, decision-making and reaction as depicted in Figure 14. In this case, CESCM as a module with the over top view of both radio and cloud side of the ecosystem will monitor cloud/radio parameters (e.g., CPU/RAM usage, call drop rate, etc.). If a violation occurs, CESCM via processing the monitoring inputs will be able to detect and then appropriately react upon the situation. The decision making process might be a simple threshold checking or a complicated multi parameter cognitive method. In the same way, the reaction ranges from the complete NS scaling, to the NS scaling up/down in/out, to the service function chain changes, to the change on a radio parameter (e.g., dedicated bandwidth to a VSCNO). For instance, assume a case where the agreed SLA with a VSCNO determines the support of a certain number of hits on an edge service. Then because of a flashy event more hits for that specific service comes, the system should be able detect the case and try to keep the SLA promise by taking an appropriate reaction.

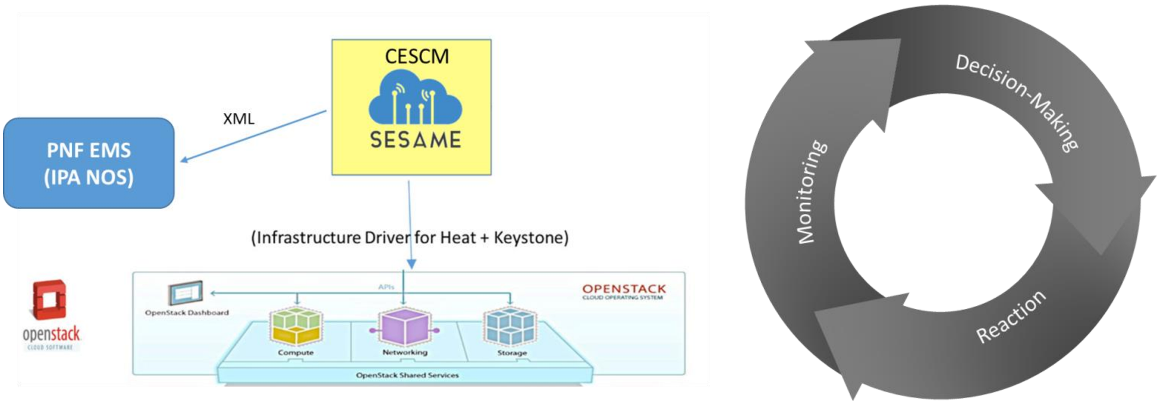

Figure 14: SESAME monitoring, decision making and reaction chain

4.1 CESC

4.1.1 Light DC micro server

The testbed environment provided at Italtel labs was been used to integrate and verify several aspects of the Light DC development:

- Hypervisor, Hardware acceleration layer and Accelerated Virtual networking (VOSYSwitch) integration on the NXP platform.
- OpenStack integration (the Kilo Release[42] was selected): computing and networking managed by OpenStack controller in case of Hybrid node (Intel/ARM).
- VNFs deployment in compute node based on:
  - Intel, CPU-only;
  - Intel, CPU+GPU;
  - Arm, CPU-only.
- VTU (Video Transcoding Unit) performance characterization in case of video transcoding service for CESCs.

The testbed is described in Figure 15.

[42] See: https://wiki.openstack.org/wiki/Kilo_Release_Schedule

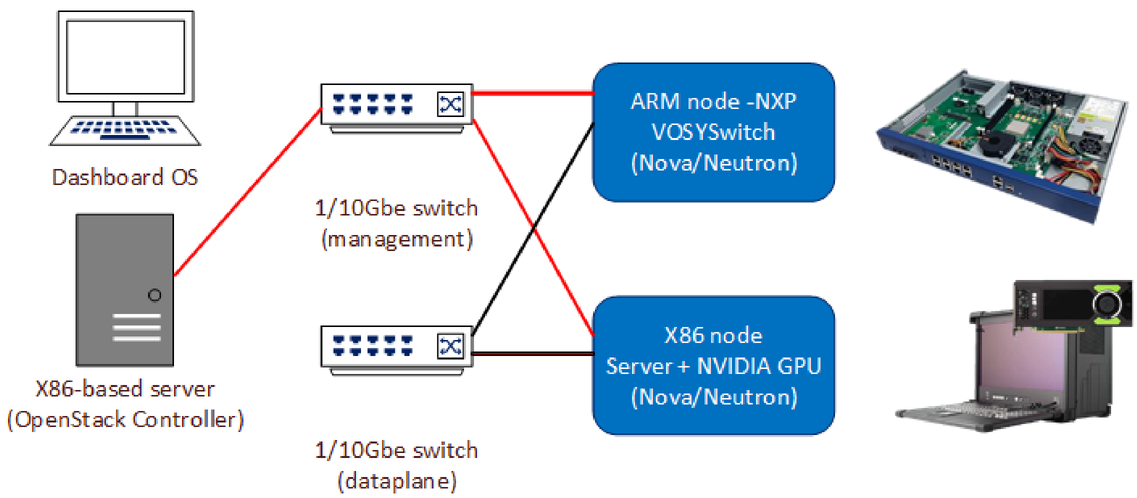

Figure 15: Micro servers for the Light DC PoC

More details can be found in [8].

4.1.2 Small Cell VNFs

4.1.2.1 SC-Common VNF

As described in section 3.1.2.1, the SC-Common VNF lies on the S1 interface between the SC PNF and the SC VNFs. It has two main purposes:

- To act as a multiplexor – de-multiplexor, routing uplink S1 messages to the appropriate SC VNF and combining downlink S1 messages into a single stream.
- To apply PNF-wide caps on the total number of UEs that may be served and the total throughput, in terms of Guaranteed Bit-Rate (GBR) radio bearers that may be established.

To date, integration testing has successfully demonstrated the following:

- That the SC-Common VNF accepts an S1 connection from the PNF.
- That the SC-Common VNF refuses an S1 connection attempt from the PNF if requires support for a PLMN that is not supported by one of the SC-VNFs.
- That the multiplexor – de-multiplexor functionality operates as intended and that S1 messages are routed to the correct VNF, according to the UE's *Selected PLMN.*
- That, as anticipated, the load place on the SC-Common VNF is very small.
- That, as intended, user plane traffic bypasses the SC-Common VNF which is only involved on RAB establishment and tear-down.

S1 connectivity and routing has been established with both Wireshark[43] traces and the successful establishment of end-to-end connectivity to the correct EPC from two test UEs with SIMs registered on different home PLMNs.

Load testing has been performed using the standard LINUX top command[44]. On a virtual machine with a single dedicated core, hosted on a PC with an Intel i7 7700 CPU[45] clocked at 3.6 GHz, the SC-Common VNF typically consumed 0.3% CPU when idle, briefly spiking to 0.7% on RAB establishment or teardown. Apart from these transitory spikes, a sustained load test of the SC VNF revealed no apparent impact on the SC-Common VNF.

PNF-*wide* caps have not been tested at this point.

4.1.2.2 SC VNF

Each instance of SC VNF handles both the signalling and user plane traffic for a single VSCNO. It connects, in the southbound direction, to the SC-Common VNF and, in the northbound direction, to an EPC that serves the VSCNO's PLMN. The SC VNF (section 3.1.2.2) implements the VSCNO's *network slice* by limiting the number of UEs that are accepted onto the cell and by constraining both the uplink and downlink throughput; non-GBR GTP-U packets in excess of the configured limits are discarded by the SC VNF.

4.1.2.2.1 Functional Integration Testing

To date, integration testing has successfully demonstrated the following:

- That the SC VNF is able to establish an S1 connection to the correct EPC and that both signalling and user plane traffic are successfully exchanged with the EPC.
- That it accepts an S1 connection from the SC-Common VNF.
- That it terminates user plane traffic in both the uplink and downlink directions.
- That the per-VSCNO UE limit works as expected. Setting the limit to one prevents a second UE from attaching to the cell. Raising the limit to any value greater than one allows a second UE to attach.
- That the above change, made by configuration management from the EMS, has more-or-less immediate effect.

The following tests have, thus far not been fully successful:

- So far, the per-VSCNO throughput caps do not appear to work. Regardless of what limit is set, the same throughput is observed. This issue is currently under investigation.

To date, the following tests have not been performed:

- Service chain testing. When service chaining is enabled, the SC VNF extracts the encapsulated payload from GTP-U packets and presents it on a dedicated Ethernet interface. On receipt of an IP packet on this interface, if the packet is successfully

[43] See: https://www.wireshark.org/
[44] See: https://linuxconfig.org/learning-linux-commands-top
[45] https://ark.intel.com/products/97128/Intel-Core-i7-7700-Processor-8M-Cache-up-to-4_20-GHz

matched to a GTP-U tunnel, it is re-encapsulated and forwarded to its destination. This functionality has not been tested at this time.

4.1.2.2.2 Load Testing

As all of a VSCNO's user plane traffic is routed via the SC VNF, it is important to understand what load this imposes and, *particularly*, whether or not the Light DC can support this load. In addition, the enabling of service chaining will, most likely, double this load as each GTP-U packet is potentially processed twice. As the SC VNF was tested on Intel platform, some extrapolation will be required.

Theoretical Maximum Throughput

Information provided by ip.access indicates that the maximum throughput of the E40 LTE Access Point (as in [15]) is approximately 75 Mbits/sec in the downlink and 25 Mbit/sec in the uplink with a single active UE and that when the uplink and downlink are active simultaneously, the aggregate throughput is approximately 85 Mbits/sec.

As additional UEs are added, total aggregate throughput decreases by approximately 10% for each doubling of the number of UEs. Thus, testing of VNF throughput can be limited to the single UE case, which can be considered the worst-case scenario.

Active UEs	**1**	**2**	**4**	**8**	**16**
Max Aggregate Throughput	85	76.5	68.9	55.8	50.2

Table 2: Approximate PNF Throughput versus Active UE Count

Test Results

A set of throughput tests were performed against a public Internet test service. Constraints in the backhaul connection to the EPC limited throughput to approximately 20 Mbits/sec in each case such that the maximum throughput of the system could not be explored. However, the following results were obtained:

	Test Run 1	**Test Run 2**	**Test Run 2**
Uplink Throughput (Mbits/sec)	17	17	18
Downlink Throughput (Mbits/sec)	18	18	19
Uplink CPU Load	26%	26%	??
Downlink CPU Load	28%	28%	??

Table 3: VNF Throughput and CPU Load

These results were obtained with a virtual machine with a single dedicated core running on an Intel i7 7700 CPU clocked at 3.6 GHz. Assuming that the performance of SC VNF is linear, these results suggest that, on the test HW platform, the SC VNF will be fully saturated with an aggregate throughput of 65 Mbit/sec.

As described in Deliverable D7.2 ([13]), the PoC Light DC has been proposed on the four HW platforms as reported in Table 1.

Given the current test results, this would indicate that**, with a single core**[46] dedicated to the SC VNF, the following *indicative* max throughput figures might be achieved:

Micro-server	Architecture	Clock Speed	Max Throughput – Service Chaining Disabled	Max Throughput – Service Chaining Enabled
Test PC	Intel i7 7700	3.6 GHz	65 Mbits/Sec	32.5 Mbits/Sec
NXP LS2085A	ARMv8, A57	1.8 GHz	32.5 Mbits/Sec	18.1 Mbits/Sec
GOMA (FlexPAC)	Xeon E5-2630v3	2.4 GHz	43.3 Mbits/Sec	21.7 Mbits/Sec
STM board	ARMv8, A53	1.3 GHz	23.5 Mbits/Sec	10.8 Mbits/Sec
Raspberry pi 3	ARMv8, A53	1.2 GHz	21.7 Mbits/Sec	10.8 Mbits/Sec

Table 4: Indicative SC VNF Downlink Throughput

The above figures need to be treated with caution, as CPU clock speed is not the only factor in determining overall throughput. Other factors include memory bandwidth, Ethernet adapter efficiency and interrupt latency.

An important fact to note is that the above figures indicate what might be achieved by a single SC VNF running on the Light DC. As each SC VNF instance will probably have its own dedicated CPU core the aggregate throughput of the PNF will increase as VSCNOs are added.

4.1.2.3 Programmable RAN sharing Functionality

The system components of the RAN sharing demonstration are shown in Figure 16. The programmable small cell is implemented through the Ettus B210[47] software-defined radio (SDR) board connected to a laptop running Ubuntu distribution 14.04[48] through a USB3 cable. The OAI software is hosted in the laptop together with OAI core network since this demonstration relied on the OAI core network software. The core network and the base station software can also be hosted on different machines optionally. As mentioned in Section 3.1.2.3, the 5G-EmPOWER eNB agent is integrated with the OAI protocol stack. Two Nexus smartphones are connected to the shared small cell: one client for each tenant. Connectivity at layer-2 is achieved with a switch

46 The SW architecture of the SC VNF allows the user plane handler to scale to utilise multiple CPU cores but this option has not been tested.
47 Please refer to: https://www.ettus.com/product/details/UB210-KIT
48 Please refer to: http://old-releases.ubuntu.com/releases/10.04.0/

that connects the machine hosting the 5G-EmPOWER VIM. A router is added for IP traffic and connectivity to the Internet. It is worth to emphasize that all components used for demonstrating RAN slicing (except switch and router) rely on open source software development.

One key component of the RAN sharing demonstration is 5G-EmPOWER VIM as discussed before. 5G-EmPOWER is a Multi-access Edge Computing OS supporting lightweight computing virtualization and heterogeneous radio access technologies. The architecture is conceptually divided into three layers. The first layer consists of the physical and virtualised resources composing the data plane. In the second layer, the 5G-EmPOWER OS is in charge of the physical and virtual resources available in the data plane. Finally, in the third layer the actual slices are managed. The entire software stack has been released under a permissive APACHE 2.0[49] for academic use and is available at the official 5G-EmPOWER website (as in [16]).

Documentation and tutorials are available in the associated GitHub project [17]; the APACHE 2.0 License regulates the commercial use of the platform.

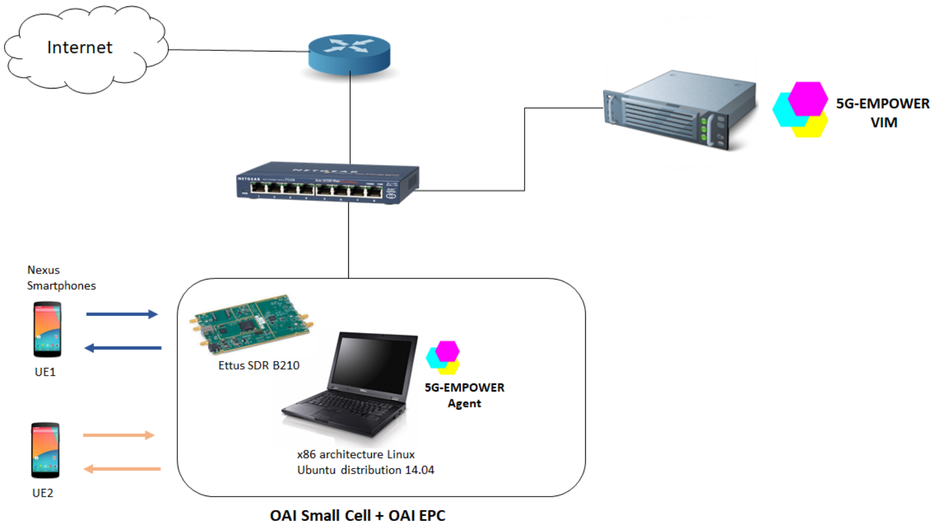

Figure 16: System components of the RAN sharing demonstration

Before showing the measurement results about the RAN sharing demonstration, it is worth to explain the working principle as shown in Figure 17. The figure shows that two tenants (e.g. SCVNOs) have assigned a slice of the radio resources (i.e. resource blocks in LTE system). The first segment represents the second tenant (T2), whereas the second the first one (T1). While no UE is receiving DL traffic in T2 network, UE1 in T1 network is receiving DL communications for a certain amount of time.

[49] https://www.apache.org/

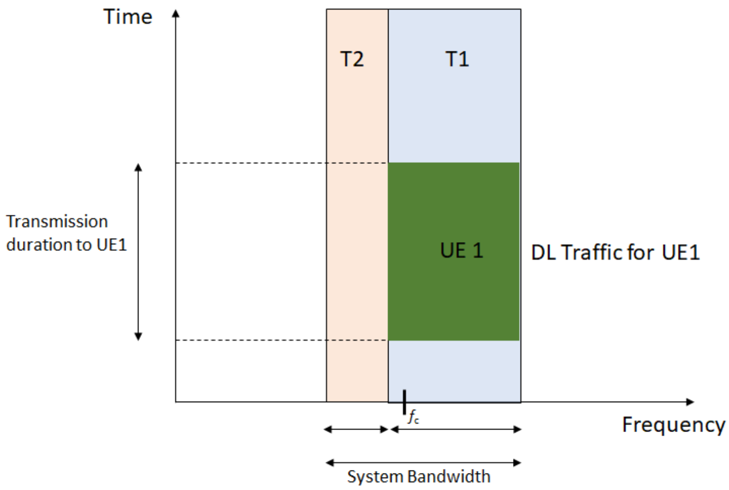

Figure 17: UEs communicating to their respective base station through RAN sharing

For the sake of showing the results, the demonstration includes an LTE small cell with an overall 5 MHz bandwidth centred at 2.660 GHz. As mentioned earlier, the first slice (that corresponds to tenant T2) and the second slice (that correspond to tenant T1) make use of two separate schedulers for scheduling UE DL traffic: Best CQI (Channel Quality Indicator) and round robin[50], *respectively*. Figure 18 shows a waterfall diagram of the downlink traffic scheduled by the two tenants. It is worth noticing that when the measurement was conducted only the UE connected to the second slice (i.e. tenant T1) is receiving downlink traffic. Furthermore, the figure shows that the RAN sharing implementation allows complete segregation of the traffic in one slice with respect to the other.

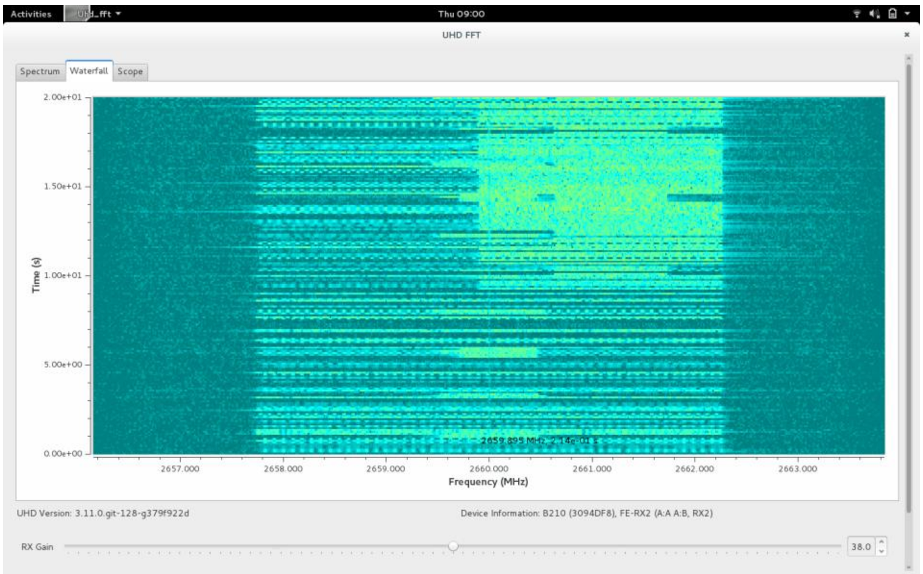

Figure 18: Measured downlink traffic in a two-slice network

50 For more informative details also see: https://en.wikipedia.org/wiki/Round-robin

4.1.3 Service VNF

4.1.3.1 vDPI

vDPI experimental results will be produced prior to Deliverable D7.4.

4.1.3.2 vWatermark

vWatermark experimental results will be produced prior to Deliverable D7.4.

4.1.3.3 vTU

Many tests were carried out to achieve a full performance characterization of the VTU, by using the testbed depicted in section 3.1.1, for the SW-only version and for the GPU-accelerated one. During all the tests it has been verified that the RAM was not completely used, to be sure that the results were not influenced by the different quantity of RAM installed in Intel-*based*, rather than ARM-*based* micro-servers.

Kind of tests:

- SW-only version of vTU on two different micro servers: ARM and x86.
- The GPU-accelerated version of VTU on the x86 micro server (equipped with NVIDIA Quadro M4000[51] GPU).

Please refer to [8] for the detailed results.

4.1.3.4 vFirewall

vFirewall experimental results will be produced prior to Deliverable D7.4.

4.1.3.5 vVideoAnalytics

Considering the PoC depicted in section 3.1.3.5, some experimental result are reported below; other detail please refer to [8] and [14].

Object tracking Use Case:

- AR service hosted at a remote server; see Figure 19.
- AR service hosted at the Light DC; see Figure 20.

[51] http://www.pny.com/nvidia-quadro-m4000

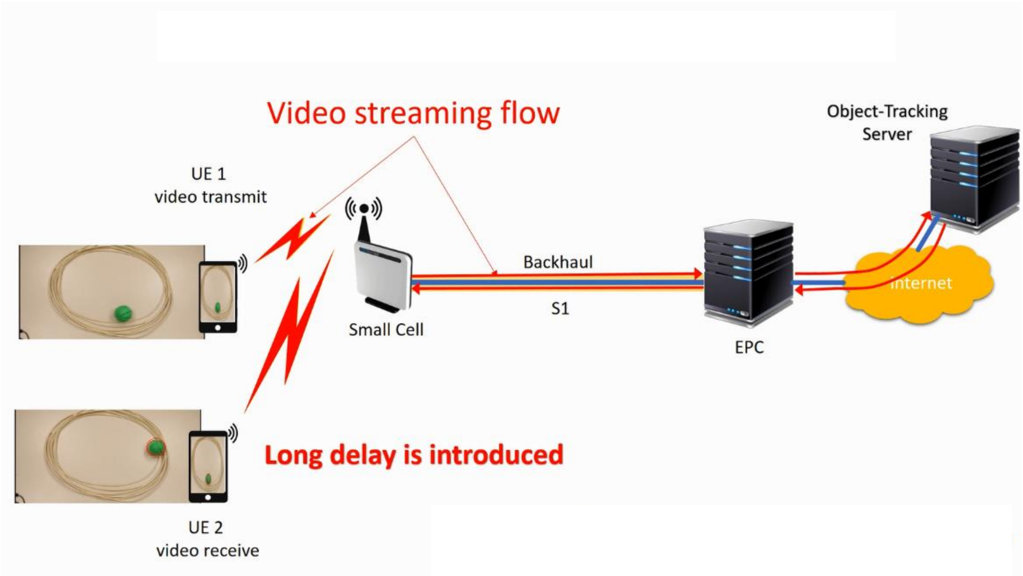

Figure 19: Object-tracking service - AR hosted at a remote server

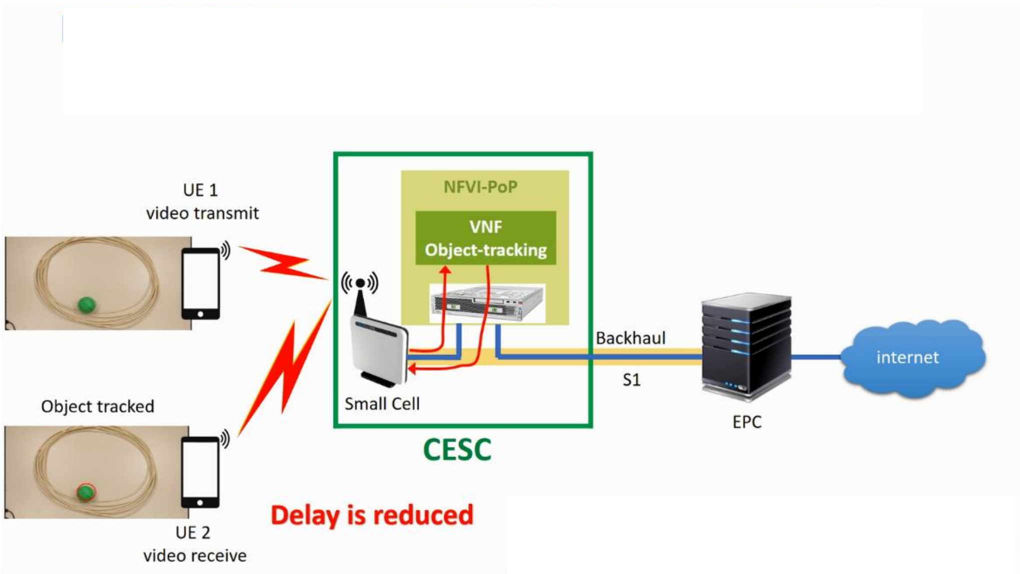

Figure 20: Object-tracking service - AR hosted at the Light DC

When a predefined object of interest moves across multiple cameras views, the pans and tilts of the available monitoring cameras are intelligently and remotely controlled by VA VNF. The continuous tracking of an object of interest can be achieved. The demonstration shows the timeliness and validity of the pan and tilt adjustment decisions made by the VA VNF and the corresponding actions taken demonstrating that edge computing can help achieve millisecond level response time to the real-time changing situations (as in Figure 21).

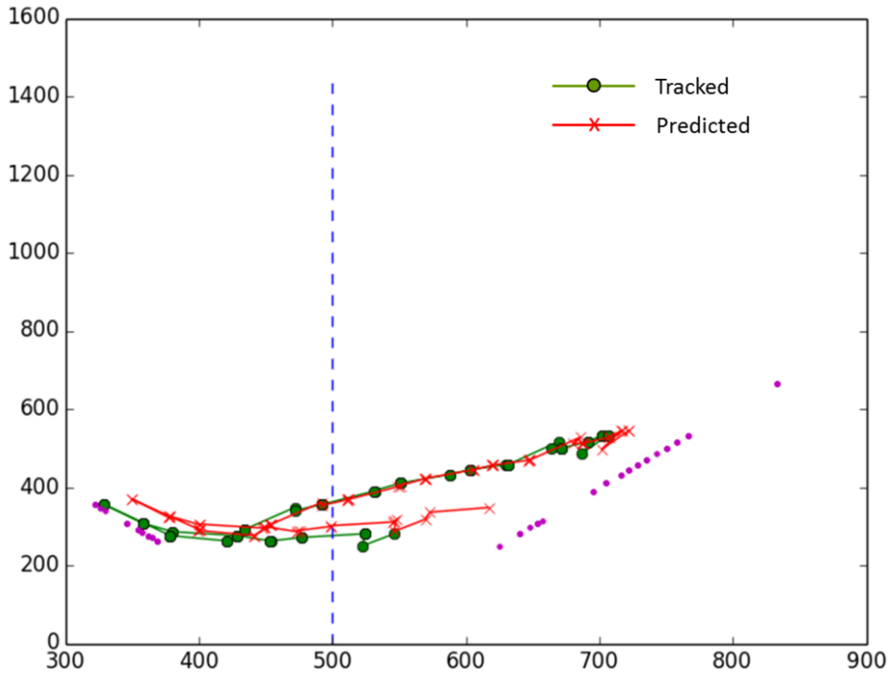

Figure 21: Tracked and predicted trajectories of a moving object

4.1.3.6 vCache

The experimental setup developed for demonstrating content caching through lightweight virtualisation technology at the edge is as shown in Figure 22. One Nexus[52] smartphone is connected to the programmable small cell. The small cell protocol stack, based on the OAI open source software, is deployed in a Docker container that runs in a laptop where Linux Ubuntu distribution 14.04 is installed. The laptop is connected to an Ettus B210 SDR through USB3 to enable the full small cell radio functions. The complete CESC is implemented connecting the small cell to a Soekris net6501[53] wherein another Docker container is deployed for the GTP encapsulation/de-capsulation function. This one requires few components, such as the Lagopus[54] software switch, the S1 interface monitor and Ryu[55] SDN controller in order to steer the traffic requests. The complete end-to-end chain is implemented by connecting the CESC to the virtualised EPC of Athonet through a VPN connection. A third Docker container is installed in a separate machine to implement the caching server. Particularly, this container deploys the Squid[56] server, which is the caching server where a content is stored. The working principle is as follows: when the user requests for a content from the Web (a web page or most likely a video content), the S1 monitor intercepts the request and upon doing the GTP traffic de-capsulation it can be checked whether the requested content is already available in the Squid caching server or not. If the content is already available, Ryu steers the request to Squid to fetch the content.

[52] See: https://www.google.com/intl/en_uk/nexus/
[53] See, *for example*: http://soekris.com/products/net6501-1.html
[54] See: http://www.lagopus.org/
[55] See: https://osrg.github.io/ryu/
[56] See: http://www.squid-cache.org/

Vice versa, if the content is requested for the first time, Ryu re-directs the request towards the EPC for standard content fetching from the Web. All Docker containers are orchestrated by the open source Docker orchestration tool Kubernetes. Three different measurements are provided to demonstrate that lightweight virtualisation by means of Docker containers is more effective in terms of resource consumption with respect to standard virtual machines implementation (the experiments were provided also in [13]). On the other hand, the demonstration of the working principle of the SESAME vCache has already been provided in [4] and in [5].

The first measurement consists of evaluating and comparing the storage space required by Docker containers with respect to a standard virtual machine deployed whereby Virtualbox[57]. This result is illustrated in Figure 23. The figure shows that the memory space occupied by the eNodeB software stack and the memory space required by Squid is remarkably less when they are deployed with Dockers (e.g. the image containing Squid on top of Ubuntu weighs 3.95 GBytes in VirtualBox, and only 358 MBytes in a Docker). On a similar basis, Figure 24 shows the CPU consumption comparing Docker containers, virtual machines deployed with Virtualbox and the native case without any virtualisation in place. The figure shows raw single core measurements for the Squid application using TOP and PERF[58]. In case of a virtual machine, this consumes 16.8% of the host CPU, whereas the Docker contain requires only 1.25%. Pinning the virtual CPU to a physical CPU further reduces the overall CPU utilization to 1.05%, which is very close value to the native system without virtualisation. The last measurement refers to the vCache boot-up time, as shown in Figure 25. This figure highlights that the Docker container implementation requires only 3.2 s, whereas the approach with standard virtual machine or even the native one require around 21 s.

To conclude, not only the working principle of content caching at the edge was demonstrated, but also that an implementation through lighter weight virtualisation technology such as dockers can be extremely effective is terms of resource consumption in a system such as the one designed within SESAME.

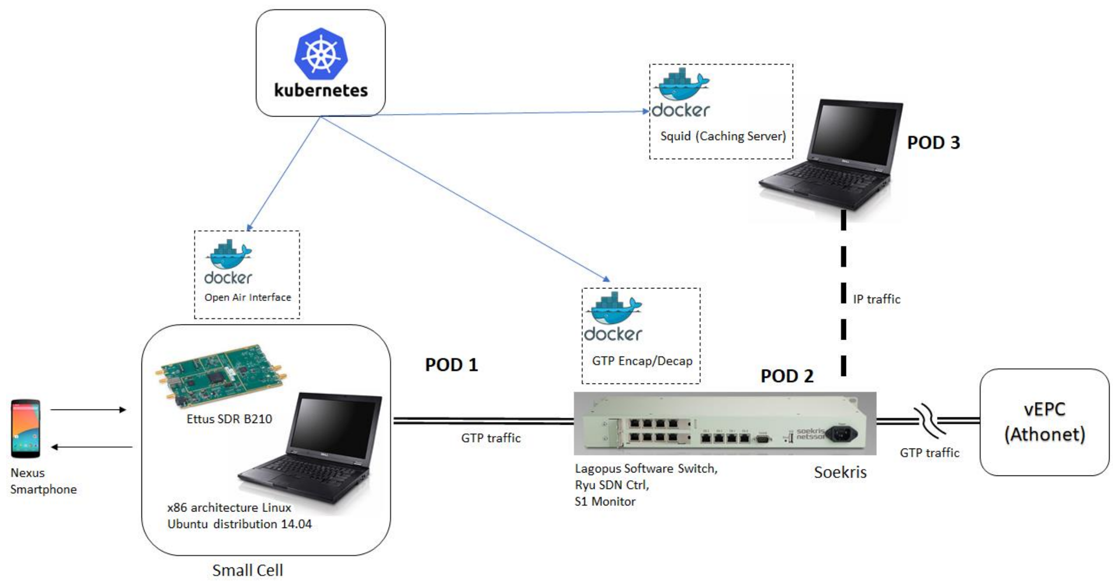

Figure 22: Experimental setup of the vCache

57 See: *https://www.virtualbox.org/*
58 For informative purpose see: *https://perf.wiki.kernel.org/index.php/Tutorial*

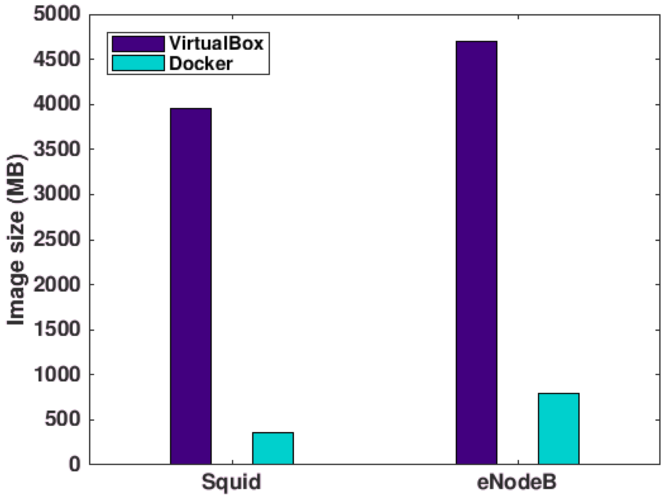

Figure 23: Measurements of storage space

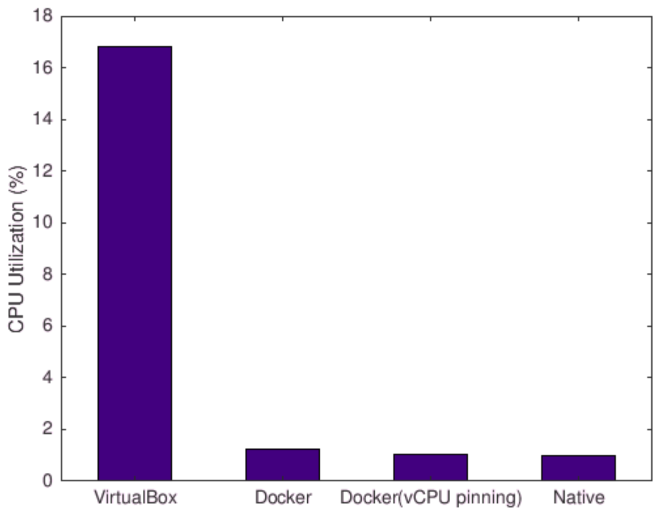

Figure 24: Measurement of CPU utilisation

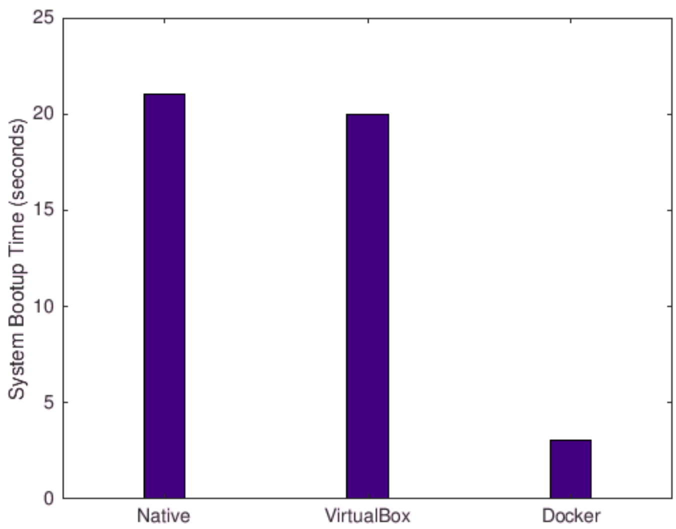

Figure 25: Measurement of the vCache booting time

4.1.4 Small Cell PNF

The Small Cell PNF E40 LTE has been extended to provide per-PLMN versions of a number of performance management counters. These are then post-processed by the EMS into separate per-VSCNO reports. In every other respect, the SC PNF is identical to the standard E40 product.

The SC PNF supports multiple PLMNs (and therefore multiple tenant VSCNOs) via a single S1 connection to the SC-Common VNF (see section 3.1.2).

To date, integration testing has successfully demonstrated the following (detailed under MOCN Testing in Deliverable D7.2 (as in [13]):

- That the SC PNF is able to be configured with multiple PLMN IDs and that these are broadcast in System Information Block SIB1.
- That UEs with SIMs having a Home PLMN containing one of the broadcast PLMNs can attach to the PNF and that UEs that do not have a supported PLMN are rejected.
- That UEs with different Home PLMNs are able to simultaneously attach to the PNF.
- That the *Reserved for Operator Use* information element can be selectively set, by configuration management, for each PLMN in the list broadcast in SIB1 and that, when set, UEs with a Home PLMN that is *reserved* do not attach to the cell.
- That the PNF is able to establish an S1 connection to the SC-Common VNF and that it correctly declares the list of PLMN IDs that it supports in the S1 SETUP request.
- That a UE's *Selected PLMN* is correctly forwarded in the TAI field of an INITIAL UE MESSAGE, allowing the SC-Common VNF to route the UE's signalling to the correct SC VNF[59].
- That data calls can be established, via the SC-Common VNF and serving SC VNF to the EPC and that bi-directional user plane traffic is exchanged.

[59] Note that user plane traffic bypasses the SC-Common VNF and is exchanged directly between SC VNF and SC PNF.

Per-PLMN performance counters (detailed under PM Testing in [13]) have not yet been tested.

To date, the EMS SLA Monitoring function has been exercised with simulated data.

4.2 CESCM

4.2.1 CESCM Dashboard

An overview of the portal's Catalog Page is provided in Figure 26.

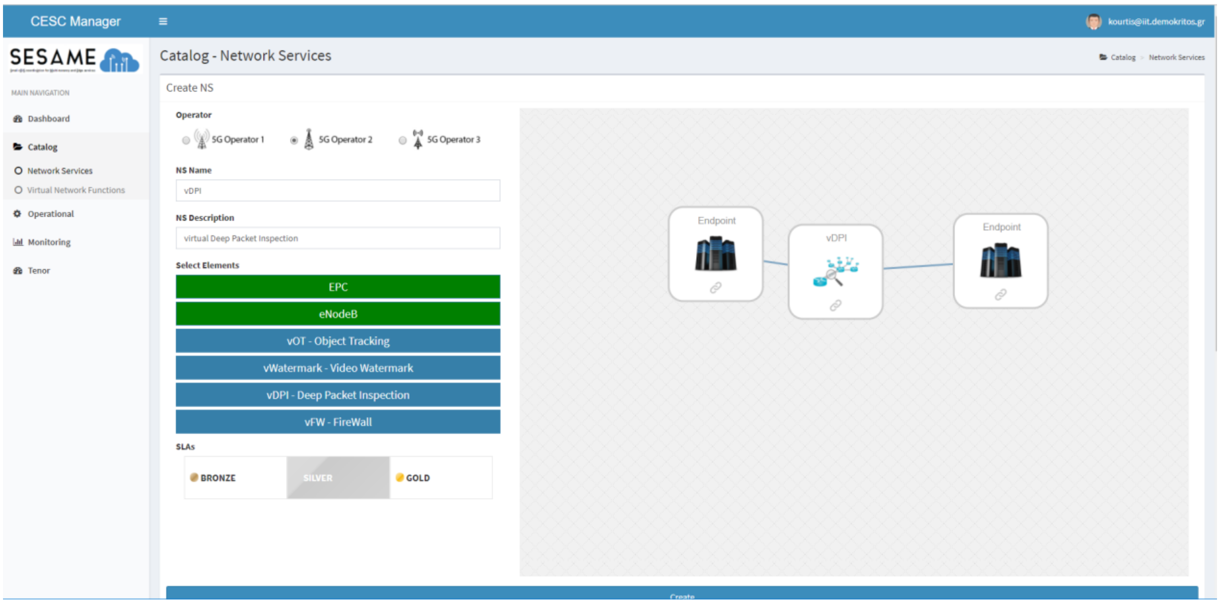

Figure 26: Portal's Catalog Page

As it can be seen, the support for multi-operator functionality has been implemented. The CESC admin, before composing a network service, can choose among a group of operators, who want to deploy their network service in the SESAME testbed.

The admin user can choose from a visualized list a predefined VNF, a set of physical resources, in our example an EPC and an eNodeB, and preview the topology of the composed service. Additionally, SLA support has been added in order to choose among a 3-tier service, which is enforced afterwards to ensure the normal operation of the provided service. This visualized topology is then translated to an NFV MANO compliant format, along with its SLA requirements.

The next steps proceed to the composition of the virtualized network service from the NFV MANO perspective and update the Monitoring service for the newly composed network service. The Monitoring service needs to be updated in order to expect alerts and SLA violations.

- Visualization of monitoring status.
- Alert notification.

Figure 27 that follows, shows the portal Operational Page.

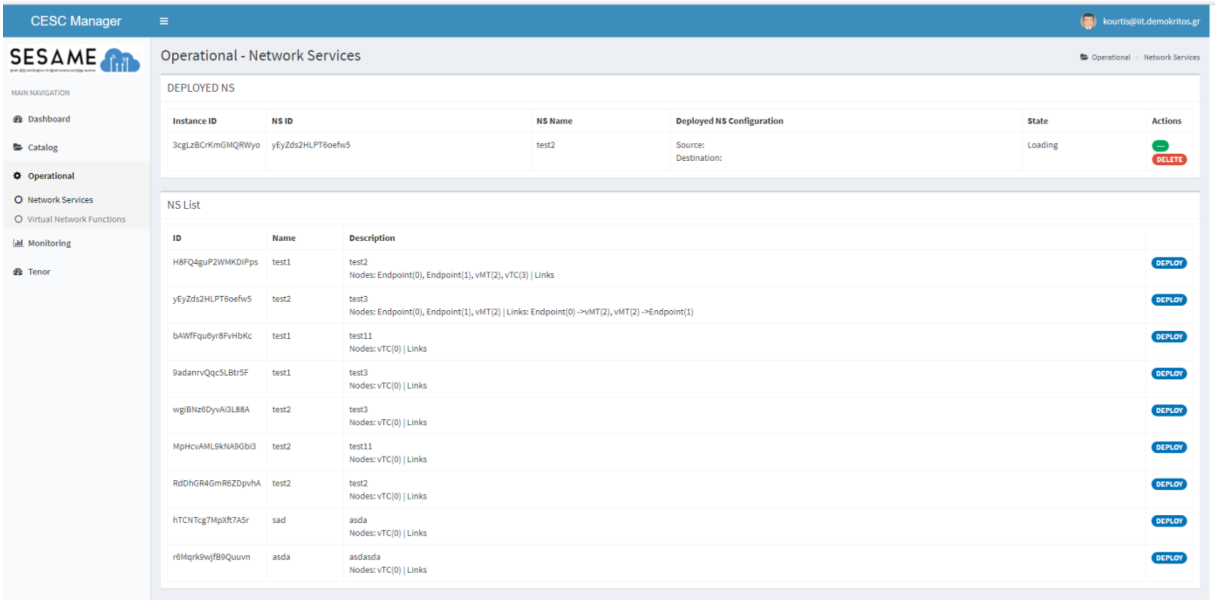

Figure 27: Portal's Operational Page

In the operational page the already composed virtualized network service, can be deployed through the NFV Orchestrator to the Light DC testbed, and afterwards initiated. In this step, the Monitoring service starts the SLA compliance supervising.

In case an SLA is violated in an initiated service, the Alert is pushed to the CESCM portal from the Monitoring service, which is then printed on the corresponding Monitoring page.

4.2.2 VIM

For the VIM PoC experimental results please refer to [10].

4.2.3 SDN Controller

In the targeted use case scenario (refer to section 3.2.3), several VNFs are spawned in the SDN-OpenStack VIM infrastructure. The idea is to perform traffic redirection to the VNFs in a particular order defined in the network service specification. The given VNFs, are vTC – that classifies the traffic according to the type (TCP, UDP) and also performs deep packet inspection. In SESAME this function performs the vDPI functionality primarily. The vWatermark adds up a watermark logo to the traffic and the vTU performs different quality adjustments to the UDP traffic being steered into the ingress port of this VNF.

There is a continuous monitoring activity performed during this process with the Netfloc's monitoring component being in charge. This component and the interfaces it has with the CESCM have been described in [10] and in [11].

The nodes where the demo pilot is running are shown in details in Figure 28 and in Figure 29. In particular, Figure 28 shows the OTE's testbed connectivity points and networks from a higher perspective, while Figure 29 magnifies the relevant ports and connections, along with the OVS bridges and connection points.

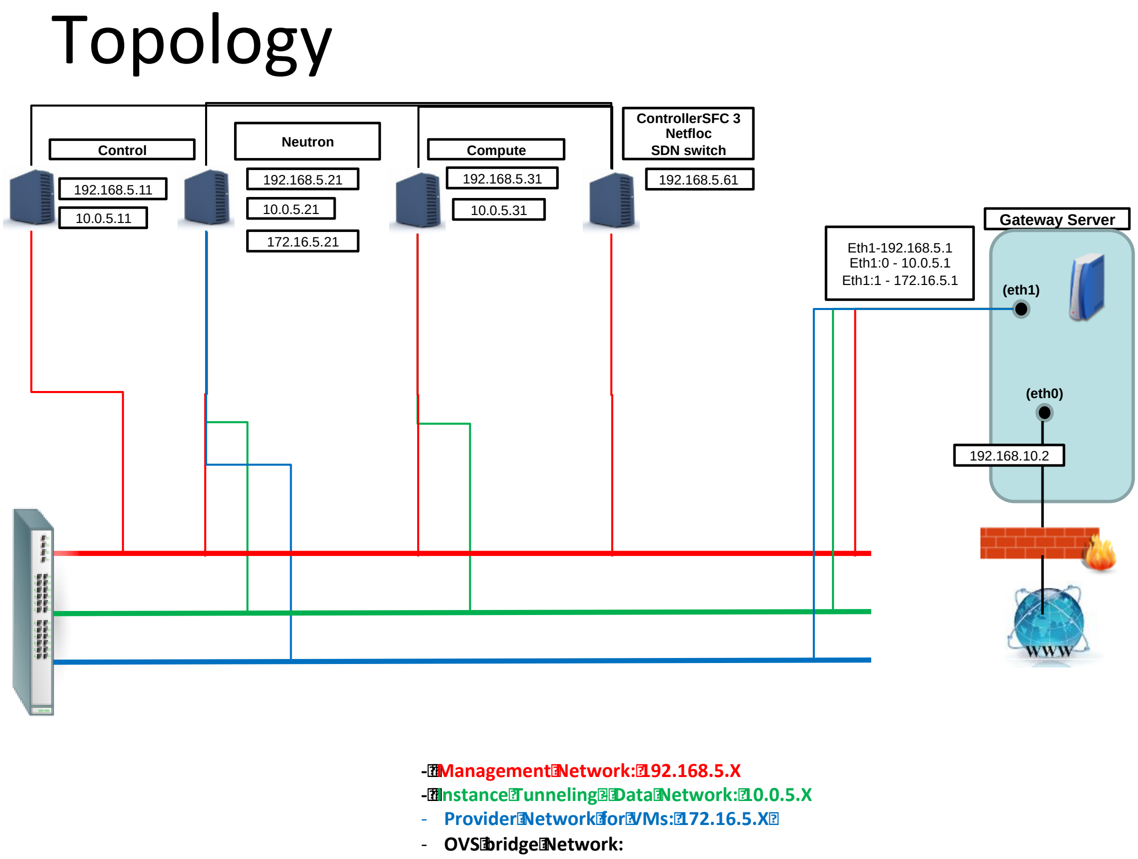

Figure 28: Overall architecture of the SESAME SDN-OpenStack OTE testbed

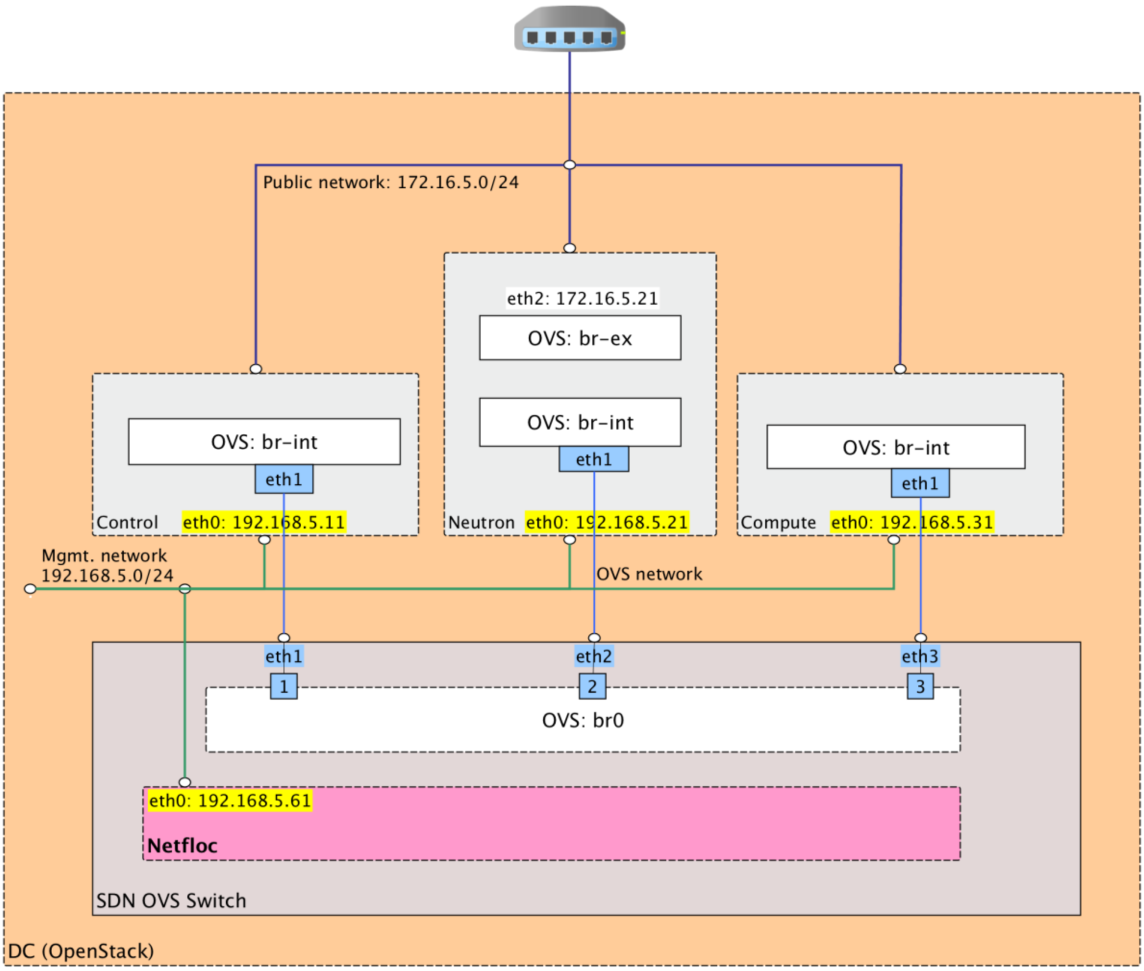

Figure 29: Detailed architecture of the OpenStack nodes and Netfloc in the SDN OTE testbed

All the nodes are running an Ubuntu 14.04.5 LTS[60] and the whole testbed is built on the x86 platform. There is additional configuration made beyond the standard OpenStack Neutron[61] configuration in order to make this testbed SDN-enabled[62]. For that matter, OpenVswitch[63] was properly configured on each of the nodes, to enable port-to-port OVS network connectivity between the br-int and br0 OVS bridges. Besides the OVS, the nodes are also connected to the public network and to additional management network, all connected via router to the OTE gateway server. After creating the OVS port, and cleaning up the Neutron DB and the MySQL DB, we proceeded on additional tests for connectivity and proper functioning of the OpenStack Neutron, Nova, and Heat[64] services. Netfloc and Netflogi were tested in order to verify correctness.

[60] Please see: http://releases.ubuntu.com/14.04/
[61] See: https://wiki.openstack.org/wiki/Neutron
[62] See: http://superuser.openstack.org/articles/open-daylight-integration-with-openstack-a-tutorial
[63] Refer to: http://openvswitch.org/
[64] For more details see: https://docs.openstack.org/heat/latest/

Later on the images of each VNFs were uploaded via the Glance ([18]), service and initial chains were deployed and tested. Finally Prometheus Docker[65] container was configured and started, with Netfloc exporter URL configured as an endpoint that serves the OpenFlow metrics. More details on the metric will be described in [10] and in [11].

An integration with TeNOR and CESCM was recently achieved and tested with a further activity to run the network service descriptors related to the chosen SFC service via TeNOR through Netfloc.

4.2.4 NFV Orchestrator

The SESAME NFVO is compliant with the ETSI NFV specification and includes the orchestration and VNFM functionality. To perform the lifecycle management of VNFs services, the NFVO will need to be integrated into the CESCM and with the VIM and SDN Controller.

The base software upon which the NFVO is being built is TeNOR[66]. The development of TeNOR exposes a REST API to perform every command in the NFVO such as on boarding VNF descriptors, deploy VNFs and get the status of network services.

Due to their popularity and simplicity, most of the software that is being used as the basis for the components in SESAME exposes a REST API. Both OpenStack and TeNOR have a well-documented northbound API through which, prior to an authentication step, a client can access each component's functions. Therefore, to make the integration process easier, REST APIs are going to be used whenever possible in the remaining pieces. The only exception to this is the SC EMS since the NOS software that is used as the basis implements the standard 3GPP SOAP interface[67] for the configuration of the Small Cell. A simple solution to solve the incompatibility of SOAP and REST[68] interfaces is to develop a proxy that is able to intercept the traffic from one of the sources and translate it to the other, although other possible solutions will be explored.

The Network Service Notification Manager (NSNM) is a feature developed in SESAME project to complete the monitoring feedback loop in practice. It is responsible of receiving and processing notifications regarding the performance of a network service deployment. NSNM is provided as a micro-service inside TeNOR but it can be also used as a standalone service communicating with the orchestrator via the API. This feature is part of the general SESAME monitoring module and only works together with Prometheus[69]. More implementation and integration details are available Iin [12].

As Figure 30 shows, the SESAME solution's user, e.g. SCNO/VSCNO system administrator, interacts with the system via the CESCM Portal developed in WP5. The portal first posts the NS and VNF descriptors to the TeNOR catalogue, where their fields are verified and stored.

Then, upon the user request, a service instantiation request is sent to trigger the deployment process. TeNOR will create the stack with the networks and machines as defined in the descriptors. In principle, a network service composed by many virtual network functions, each of which delivering a certain functionality. The bigger the service is the most important monitoring

65 See: https://hub.docker.com/r/prom/prometheus/
66 See: https://github.com/T-NOVA/TeNOR
67 Also see: https://en.wikipedia.org/wiki/SOAP
68 Also see: https://en.wikipedia.org/wiki/Representational_state_transfer
69 Also see: https://prometheus.io/

gets. When all the monitoring related modules have been set correctly, upon the NS instantiate TeNOR will take care of two main actions:

- Start the monitoring data exporter inside every machine deployed by the service;
- Register the service for monitoring in the Prometheus module that will start collecting all the critical info about the running VNFs;

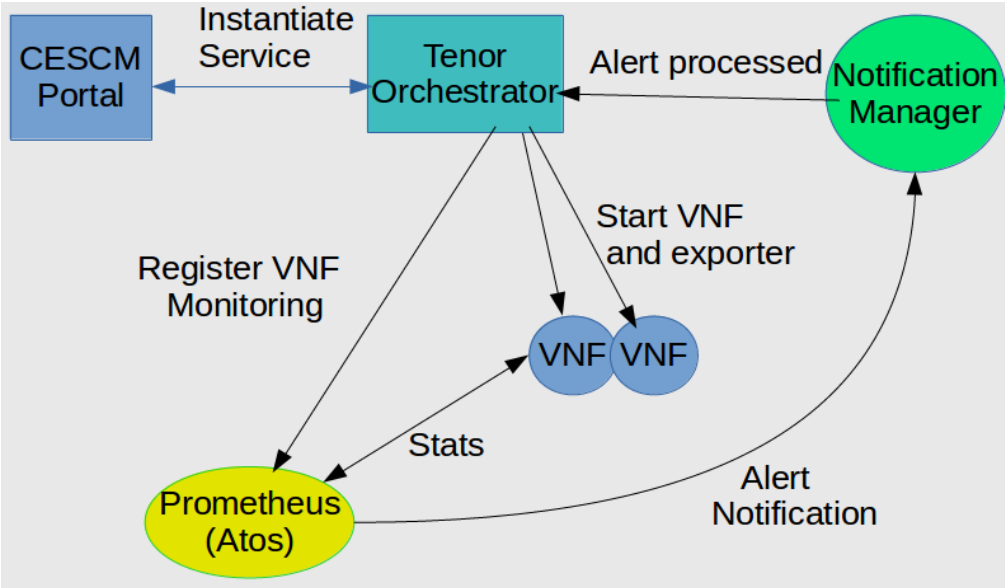

Figure 30: SESAME Monitoring Schema

As stated above, NFVO is the responsible module to perform the appropriate reaction mechanism upon receiving monitoring alarms. Such a reaction mechanism might be as simple as restarting a service to more complicated actions such as migrating a service, scaling it up and down, etc.

The integration of TeNOR with the rest of components is done very easily. Figure 31 below shows a screen of TeNOR software where the OpenStack API endpoint can be specified to be used as the VIM. Under the "configuration" dropdown, there is an option to introduce a Point of Present (PoP) which corresponds to a VIM for TeNOR (it is worth noting that in general TeNOR is able to manage more than one VIM simultaneously). By clicking on the PoP, the dialog below will appear. The most important item on the dialog is the OpenStack IP which determines where OpenStack is located. TeNOR supports OpenStack Juno[70] version or higher and both Keystone[71] v2.0 and v3.0. After the introducing the OpenStack, TeNOR will assign a PoP ID to it. Since for the SESAME prototype only one VIM is considered, the assigned ID will represent the PoP where all services of all tenants (VSCNOs) will be instantiated (i.e.: the Light DC).

[70] See: https://releases.openstack.org/juno/
[71] See: https://releases.openstack.org/juno/#keystone

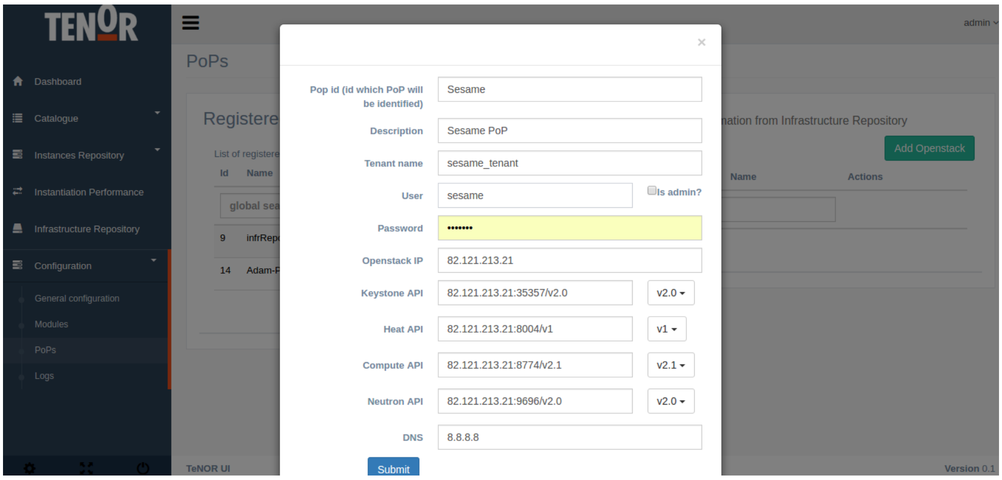

Figure 31: How to make TeNOR communicate with OpenStack

TeNOR GUI has included a "configuration" dropdown which makes the configuration process easy and user friendly. Besides introducing OpenStack to TeNOR (as explain above), it is possible to assign credentials and access right to the TeNOR users (in SESAME, both SCNO and VSCNO). In the context of SESAME, this is an important configuration step, since the view of SCNO and VSCNOs are different. While SCNO has an overview of all available resources and running services, a VSCNO should have access only on its own services. TeNOR perfectly is able to carry out such separation/isolation.

4.2.5 EMS

To date, integration testing of the EMS has successfully demonstrated the following:

- That it is possible to create the SESAME specific CESC and VSCNO managed object sub-trees.
- That, once provisioned as part of a CESC, the associated SC PNF, SC-Common VNF and each SC VNF establishes a management connection to the EMS allowing its configuration to be changed and it to report any faults that it encounters.
- That the "Create Virtual Cell" EMS wizard operates as expected:
    - It allows the user to select a "Provisioned SLA" that defines the network slice that they require.
    - It allows the user to select a host CESC from those available that has sufficient spare capacity for to support the VSCNO's selected network slice and does not already host a virtual cell with the VSCNO's PLMN ID.
    - Once a host CESC is selected, the wizard completes configuration of the SC PNF, SC-Common VNF and SC VNF in order to support the VSCNO's virtual cell and network slice.
- That it is possible, by means of EMS user permissions, to add SCNO and VSCNO users to the EMS such that:
    - The SCNO, as infrastructure owner, is able to view and modify all managed objects as required.

   - That each VSCNO is only able to view and modify the managed objects that are part of their own virtual network and have no visibility of the managed objects that are the responsibility of the SCNO or another VSCNO.

- That it is possible to configure Monitored SLA objects with a defined list of monitored KPIs, thresholds and action to take on KPI threshold crossing. That these SLAs may be applied to either all of the virtual cells belonging to the VSCNO or a list of specific virtual cells.
- That KPI values calculated by the EMS, based on performance management reports from the PNF, can be extracted from the EMS database and forwarded to the CESCM in JSON format.
- That changes to the configuration of an SC VNF can be achieved via one of the northbound configuration management interfaces.

To date, the following EMS tests have not been fully successful:

- When a VSCNO administratively locks a virtual cell object, the virtual cell should be taken out of service by marking the VSCNO's PLMN ID as *reserved for operator use* in the SC PNF. This does not appear to operate reliably.
- Monitored SLA objects do not appear to be correctly evaluating KPI values and are generating incorrect SLA Breach alarms.

4.2.5.1 Knowledge Discovery capabilities in the EMS

This section illustrates some initial experimental results to validate the behaviour of the knowledge discovery capabilities described in section 3.2.5.1 for the energy saving use case.

The considered scenario considers three different real small cell deployments. The PM files of the first deployment include nine different small cells belonging to an operator providing service on an island in the Pacific Ocean. The cells were deployed mainly in office blocks, hotels and the residences of VIPs. Whilst hand-in and hand-out to the macro network was possible, the small cells did not perform handovers to other small cells. The second deployment includes one small cell belonging to a national operator in a central European country. It is deployed as stand-alone cell in a shop belonging to the operator and, typically, did not perform hand-overs to any other cells. The third deployment includes 23 small cells. They belong to an operator providing service on an island in Northern Europe and were used to provide service mainly to users in their homes, in public houses and restaurants. Whilst hand-in and hand-out to the macro network was possible, the small cells did not perform handovers to other small cells.

The available PM files for the considered small cells include the metrics for a total of one day. Then, the pre-processing stage shown in Figure 10 builds, for each cell, a time series X_i composed of M=24 samples with the hourly values of the traffic in the cell. As for the training set, it consists of a total of S =228 cells from the deployment of [22] .

The results presented in this deliverable are obtained for the decision tree classifier, while a more detailed comparison among the different classifier types considering all the cells of the abovementioned deployments will be provided in the forthcoming deliverable D7.4.

Figure 32 depicts the resulting decision tree classifier that has been learnt after the training stage with the cells of the training set. In turn, Figure 33 shows the time domain traffic pattern

of two of the abovementioned small cells classified as A (candidate cell to be switched off) and B (cell that cannot be switched off) by this decision-tree classifier. Looking at these time domain traffic patterns this classification appears as an adequate decision because the cell classified as A exhibits relatively long periods at night serving no traffic at all while the cell classified as B exhibits traffic during most of the time.

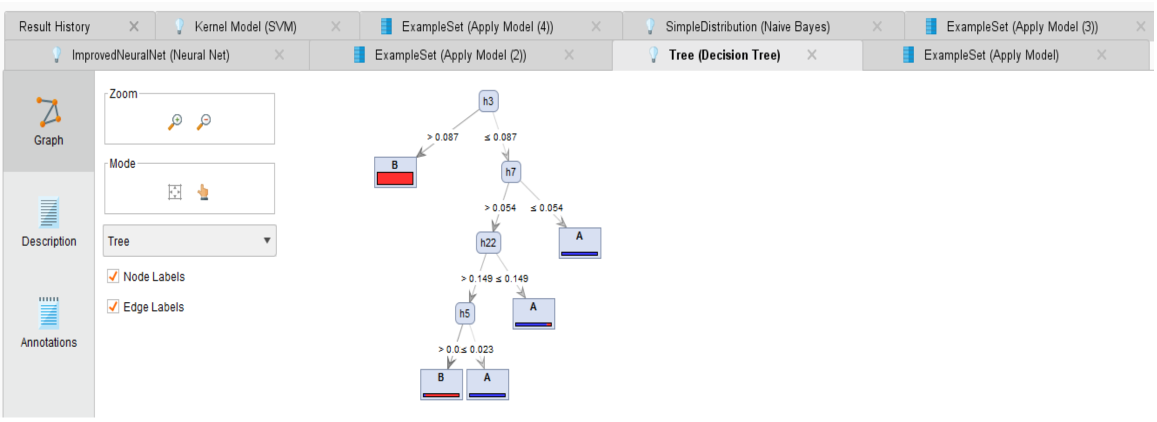

Figure 32: Decision tree classification model

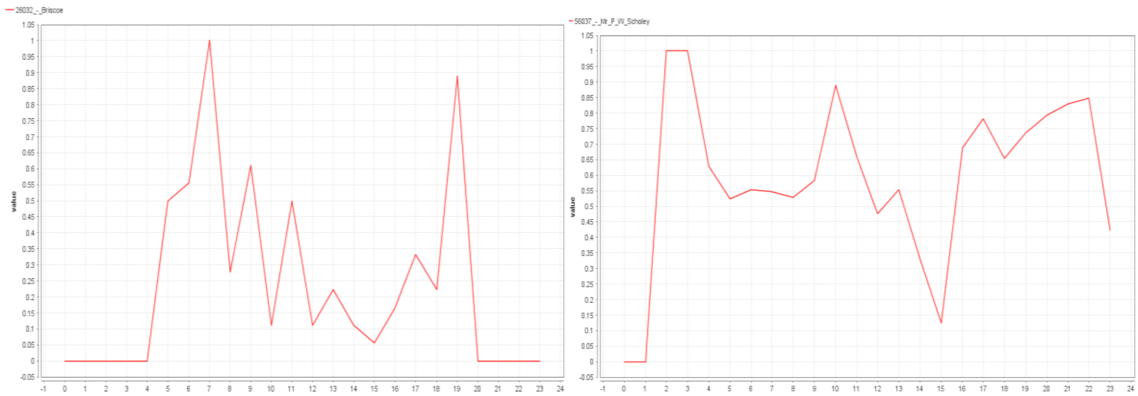

Figure 33: Example of the time domain pattern of a cell classified as A (Left) and B (Right)

4.2.6 SLA monitoring

As mentioned before, the SLA monitoring output is presented in two types of notifications: soft threshold and hard threshold alarms. This notification differs from the severity of the notification, in the first case; the system is still able to reallocate the resources in order to provide the expected quality before the agreement is broken. The hard threshold alarm, depicts the violation of a services contracted, this violation will need to analyse further the

infrastructure, as the action provided in the infrastructure is not enough to provide the service quality.

Soft threshold alarms are forwarded to the management system responsible of the infrastructure, the message is represented as follow:

```
{
  "status": "firing",
  "groupLabels": {
    "service": "demo-service",
    "alertname": "instance_downn"
  },
  "groupKey": {},
  "commonAnnotations": {
    "summary": "Monitor service non-operational"
  },
  "alerts": [
    {
      "status": "firing",
      "labels": {
        "alertname": "instance_downn",
        "service": "demo-service",
        "instance": "localhost:8088",
        "job": "POP",
        "env": "prod",
        "service_id": "hb323kl789688g",
        "severity": "critical"
      },
      "endsAt": "0001-01-01T00:00:00Z",
      "generatorURL": "http://prometheus:9090/graph?g0.expr=up+%3D%3D+0&g0.tab=0",
      "startsAt": "2017-05-22T18:13:42.822+02:00",
      "annotations": {
        "description": "localhost:8088 service is down.",
        "summary": "Monitor service non-operational"
      }
    }
  ],
  "version": "4",
  "receiver": "tenor-notification",
  "externalURL": "http://prometheus:9093",
  "commonLabels": {
    "alertname": "instance_downn",
    "service": "demo-service",
    "job": "POP",
    "env": "prod",
    "service_id": "hb323kl789688g",
    "severity": "critical"
  }
}
```

The notification of the violation forwarded to the CESCM portal as follow:

```
&lt;violation&gt;
   &lt;uuid&gt;2396c9f3-bdaf-4305-8be4-2b137054642c&lt;/uuid&gt;
   &lt;agreement_id&gt;agreement-sesame&lt;/agreement_id&gt;
   &lt;service_name&gt;sesame-service&lt;/service_name&gt;
   &lt;service_scope&gt;&lt;/service_scope&gt;
   &lt;kpi_name&gt;availability&lt;/kpi_name&gt;
   &lt;datetime&gt;2017-07-27T14:22:30+02:00&lt;/datetime&gt;
   &lt;actual_value&gt;0.6459022667315054&lt;/actual_value&gt;
&lt;/violation&gt;
```

This information is depict in the portal for the user information, so as the consumer and the provider can agree on the compensation due to this violation.

5 Conclusion

This document describes PoCs, testbeds and experimental integration results of the HW and SW modules developed during the SESAME project, in order to realize all the functional blocks envisaged by the architecture.
Therefore, throughout the document can be found references to deliverables closed in the packages related with the technical objectives of the project (WP3, WP4, WP5 and WP6) which describe the outcomes of related carried out activities.

The deliverable will be used as “track” for the overall integration phase of the SESAME platform that will be depicted in the Deliverable D7.4.

6 References

[1] SESAME Deliverable D2.2: "*Overall System architecture and Interfaces – First Iteration*", H2020 SESAME project, *March 2016*.

[2] SESAME Deliverable D2.3: "*Specification of the CESC components*", H2020 SESAME Project, *March 2016.*

[3] SESAME Deliverable D3.1: "*CESC Prototype design specifications and initial studies on Self-X and virtualization aspects*", H2020 SESAME project, *June 2016.*

[4] SESAME Deliverable D3.3: "*Framework of a distributed network management system capable to host and run Self-X*", H2020 SESAME project, *June 2017.*

[5] SESAME Deliverable D3.4: "*CESC Small Cell prototype and PoC*", H2020 SESAME Project, *June 2017.*

[6] SESAME Deliverable D4.1: "*Light DC architecture design*", H2020 SESAME Project, *June 2016*.

[7] SESAME Deliverable D4.2: "*Virtualization extensions for acceleration of Light DC capabilities*", H2020 SESAME project, *December 2016*.

[8] SESAME Deliverable D4.3: "*Techniques for efficient VNF Deployment with relevant VIM extensions, Evaluation framework*", H2020 SESAME project, *June 2017.*

[9] SESAME Deliverable D4.4: "*Light DC prototype*", H2020 SESAME project, *June 2017.*

[10] SESAME Deliverable D5.2: "*VIM and CESCM implementation*", H2020 SESAME project, *September 2017.*

[11] SESAME Deliverable D5.3: "*Techniques and optimisation of VNF placement algorithms – Security issues*", H2020 SESAME project, *September 2017.*

[12] SESAME Deliverable D6.3: "*Service Management and Orchestration functions, including VNF models*", H2020 SESAME project, *September 2017.*

[13] SESAME Deliverable D6.4: "*Orchestrator Prototype*", H2020 SESAME project, *September 2017*

[14] Sesame Deliverable D7.2: "Integrated CESC Prototype Validation", H2020 SESAME project, *July 2017*

[15] *http://www.ipaccess.com/uploads/wysiwyg_editor/files/2017/E40-Datasheet-v1.0.pdf*

[16] 5G EmPOWER web site: *http://empower.create-net.org/*

[17] 5G EmPOWER GitHub project: *https://github.com/5g-empower/5g-empower.github.io/wiki*

[18] OpenStack wiki for Glance: *https://wiki.openstack.org/wiki/Glance*

[19] Wiki page: *https://en.wikipedia.org/wiki/Logical_Volume_Manager_(Linux)*.

[20] 3GPP TS 32.435 v14.0.0, "Performance measurement; eXtensible Markup Language (XML) file format definition (Release 14)", *April 2017.*

[21] RapidMiner Studio, *http://www.rapidminer.com*

[22] J. Pérez-Romero, J. Sánchez-González, O. Sallent, R. Agustí (2016): On Learning and Exploiting Time Domain Traffic Patterns in Cellular Radio Access Networks. In *Proceedings of the 12th International Conference on Machine Learning and Data Mining (MLDM)*, pp.501-515, New York, US, *July 16-21, 2016*. Springer, New York.

[23] R.A. Wilson and F.C. Keil (1999): *The MIT Encyclopedia of the Cognitive Sciences*. MIT Press.