Grant Agreement N°: 957317
Topic: ICT-42-2020
Type of action: IA

High-tech and affordable 5G network roll-out to every corner

D4.1: Integration and Affordable5G roll-out plans

Revision: v1.0

Work package	WP 4
Task	Task 4.1
Due date	31/08/2021
Submission date	31/08/2021
Deliverable lead	UMA
Version	1.0

Abstract

The goal of this deliverable is to define the guidelines and procedures to integrate the building blocks reported in deliverables D2.1 and D3.1 to become part of the two 5G testbed platforms used for the demos and trials in the project. The integration plan defines how to map the Affordable5G architecture to Castellolí and Malaga 5G platforms and the specific timing for such integration. The key idea is to define several test cases to ensure that the blocks are properly working in the target platforms. This document provides a first version of test cases that will be expanded and implemented in the second part of the project. Deliverable D4.2 will report the final list of test cases and the results of the test campaigns with the final Affordable5G platforms.

Keywords: 5G Non-Public Networks, O-RAN, NFV, 5GC, AI/ML-based network optimization, System architecture, integration methodology, testing methodology.

List of Contributors

Partner	Short name	Contributor(s)
ATOS SPAIN SA	ATOS	Sergio Gonzalez, Borja Otura and Josep Martrat
ADVA Optical Networking Israel Ltd	ADVA	Andrew Sergeev
RETEVISION I SA	CEL	Judit Bastida Raja and Raul Gonzalez Prats
ACCELLERAN	ACC	Simon Pryor
ATHONET SRL	ATH	Marco Centenaro, Nicola di Pietro, Arif Ishaq, and Daniele Munaretto
THINK SILICON EREYNA KAI TECHNOLOGIA ANONYMI ETAIRIA	THI	Georgios Keramidas
RUNEL NGMT LTD	REL	Israel Koffman and Baruch Globen
NEMERGENT SOLUTIONS S.L.	NEM	Marta Amor and Aarón Rodríguez
UBIWHERE LDA	UBI	Luís Conceição and Juliana Teixeira
MARTEL GMBH	MAR	Gabriele Cerfoglio and Giacomo Inches
EIGHT BELLS LTD	8BELLS	George Kontopoulos
NEARBY COMPUTING SL	NBC	Oscar Trullols and Angelos Antonopoulos
UNIVERSIDAD DE MALAGA	UMA	Pablo Herrera Díaz, Javier Andrés Jiménez Jiménez, Adrián Pérez Aguilar, Francisco José Luque Schempp, Bruno García García and Pedro Merino Gómez
ETHNIKO KAI KAPODISTRIAKO PANEPISTIMIO ATHINON	NKUA	Panagiotis Trakadas, Lambros Sarakis, Panagiotis Gkonis, Sotirios Spantideas and Anastasios Giannopoulos
FUNDACIO PRIVADA I2CAT, INTERNET I INNOVACIO DIGITAL A CATALUNYA	I2CAT	Estefanía Coronado, Wilson Ramirez and Giovanni Rigazzi,
EURECOM	EUR	Navid Nikaein

Document Revision History

Version	Date	Description of change	List of contributor(s)
V0.0	08/04/2021	ToC	UMA
V0.1	10/06/2021	New information added in section 3, section 4 and section 6	UMA, ATH, NKUA, ATOS, NEM
V0.2	17/06/2021	New information added in section 3, I2CAT includes their contribution in section 4 and some test cases are defined	UMA, I2CAT
V0.3	27/06/2021	Section 3 is completed, contributions of THI, ACC, EUR, REL and ADVA to section 4 are included	UMA, THI, ACC, ADVA, EUR, REL
V0.4	01/07/2021	Contributions of NBC, MAR and NEM to section 4 are included	NBC, MAR, NEM
V0.5	16/072021	Introduction is included and section 6 is updated	UMA
V0.6	20/07/2021	UBI contribution to section 4 is included. Section 5 is completed by CEL	UBI, CELL
V0.7	23/07/2021	Executive summary and conclusion are included, and new section 7 is added to group all test cases. References, titles, and other minor errors are corrected	UMA
V0.8	23/08/2021	Modification of some parts of the document after revision by NKUA and ATOS	UMA, NKUA, ATOS
V0.9	27/08/2021	Contribution to Section 3, description of the slice manager and non-RT RIC	I2CAT, UMA
V1.0	31/08/2021	Final check for submission	ATOS

Project co-funded by the European Commission in the H2020 Programme		
Nature of the deliverable:		R
Dissemination Level		
PU	Public, fully open, e.g., web	√
CI	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to Affordable5G project and Commission Services	

EXECUTIVE SUMMARY

Affordable5G is addressing the construction of end-to-end non-public 5G networks (5G NPNs) based on hardware and software components provided outside the mainstream of the big full packaged commercial solutions. The project is producing several building blocks to be integrated in the 5G platforms already available in Castelloli and Malaga, which are the locations where the Affordable5G is developed and deployed. Such blocks include features like Open RAN, core network, transport, slicing, orchestration, time sensitive networking, as well as services for Mission Critical Communications and Real time video analytics in cities.

The deliverable describes the approach to the integration and testing of selected blocks per platform to make two instantiations of the general architecture designed in the project. This process is defined with two main ideas: i) an incremental integration of the new components in the target platforms and ii) a rigorous automated testing based on test cases. The main novelties are related to the testing methodology, that adapts the global consensus in the 5GPPP TMV WG to the specific requirements of Affordable5G. The project has selected the OpenTAP approach used in 5GENESIS project to implement the testing methodology and defines in this deliverable an initial set of test cases involving most of the building blocks.

The plan is described with a detailed scheduling that will be revised and updated until the production of the next version of this deliverable at the needs of the project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

3GPP	Third Generation Partnership Project
4G	Fourth Generation
5G	Fifth Generation
5GC	5G Core
ADB	Android Debug Bridge
AI	Artificial Intelligence
AF	Application Function
AGV	Automated Guided Vehicle
AMF	Access and Mobility Function
AMR	Autonomous Mobile Robots
AUSF	Authentication Server Function
BSS	Business Support System
CAPIF	Common API Framework
CI	Continuous Integration
C-MDAF	Centralized Management Data Analytics Function
CN	Core Network
CP	Control Plane
CU	Central Unit
CPRI	Common Public Radio Interface
DA	Data Analyzer
DC	Data Collector
DL	Deep Learning
DN	Data Network
DRL	Deep Reinforcement Learning
DSCP	Differentiated Services Code Point
DS	Data Source
DSF	Data Semantic Fabric
DP	Data Plane
DU	Distributed Unit
E2E	End to End
eMBB	enhanced Mobile Broadband
EMS	Element Management System
EMS-CM	EMS - Configuration Management
ENI	Experiential Network Intelligence
FEC	Forward Error Correction
FoF	Factory of the Future
GPU	Graphics Processing Unit
IM	Infrastructure Monitoring
IMF	Infrastructure Management Framework
IoT	Internet of Things
IP	Internet Protocol
ITU	International Telecommunication Union
KNF	Kubernetes-based Network Functions
KPI	Key Performance Indicator
LiFi	Light Fidelity
LLS	Lower Layer Split
LSTM	Long Short-Term mMemory
MANO	Management and Orchestration
MBO	Mobile Backhaul Orchestrator
MCS	Mission Critical Service
MCPTT	Mission Critical Push-To-Talk
MEC	Multi-access Edge Computing
ML	Machine Learning

mMTC	massive Machine Type Communications
MNO	Mobile Network Operator
MOCN	Multi-Operator Core Networks
MORAN	Multi-Operator RAN
NBI	Northbound Interface
NF	Network Function
NSMF	Network Slice Management Function
NSSMF	Network Slice Subnet Management Function
NFL	Network Function Layer
NFV	Network Function Virtualization
NN	Neural Network
NPN	Non-Public Network
NR	New Radio
NST	Network Slice Template
NWDAF	Network Data Analytics Function
OAM	Operations, Administration and Management
O-CU	Open Central Unit
O-DU	Open Distributed Unit
ONAP	Open Network Automation Platform
O-RAN	Open Radio Access Network
O-RU	Open Remote Unit
OSM	Open Source MANO
PCC	Policy Charging and Control
PCF	Policy Control Function
PDU	Protocol Data Unit
PLC	Programmable Logical Controller
PNF	Physical Network Function
PSF	Physical Security Function
PLMN	Public Land Mobile Network
PNI-NPN	Public Network Integrated – Non-Public Network
PM	Performance Monitoring
PPDR	Public Protection and Disaster Relief
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technology
RIC	Radio Intelligent Controller
RIS	Reconfigurable Intelligent Surface
RL	Reinforcement learning
RNN	Recurrent Neural Network
RoE	Radio Over Ethernet
RRM	Radio Resource Management
RT	Real Time
RU	Remote Unit
SA	Standalone
SBA	Service Based Architecture
SDN	Software Defined Networking
SEAL	Service Enabler Architecture Layer
SLA	Service Level Agreement
SME	Small of Medium Enterprise
SMF	Session Management Function
SMO	Service Management and Orchestration
SNPN	Stand-alone Non-Public Network
SON	Self Organizing Network
SUT	System Under Test
TCP	Transmission Control Protocol
TNE	Transport Network Equipment

TSN	Time Sensitive Networking
UAV	Unmanned Aerial Vehicle
UDM	Unified Data Management
UDR	Unified Data Repository
UE	User Equipment
UMa	Urban Macrocells
UP	User Plane
UPF	User Plane Function
UE	User Equipment
URLLC	Ultra Reliable Low Latency Communication
V2X	Vehicle to Everything
VDU	Virtual Deployment Units
vEPC	virtualized Evolved Packet Core
VNF	Virtual Network Function
VoMS	Vertical-oriented Monitoring System
VSNF	Vertical Service Management Function
VSF	Virtual Security Function
WDM	Wavelength Division Multiplexing

1 INTRODUCTION

The development of an affordable 5G system, which is the main objective of this project, is based on the combination of many building blocks that, together, conform a full functional structure. To achieve this, one of the most critical steps is the integration and testing of the components. This deliverable is specifically focused on that point.

Before discussing the structure of the deliverable, this is a brief presentation of the new building blocks to be developed:

- NEOX^{TM} [1] accelerator, which is a low power consumption hardware accelerator based on a multicore and multithread GPU architecture.
- 5GCORE, which is a SA fully virtualized 5GC.
- O-RAN (Open Radio Access Network), which forms the radio access part of the architecture and is the combination of the following building blocks: O-CU, O-DU, O-RU and Near-RT RIC (dRAX).
- AI/ML Framework, which provides Affordable5G with the capability to integrate AI and ML algorithms in models to infer in control loops of other building blocks.
- MANO for services. Two different solutions of e2e orchestrators are developed in the project.
- Telemetry, which consists of real time data collection from various building blocks of the architecture.
- TSN over 5G. Two different approaches are proposed in the project, which plan to build a wireless time sensitive network using Affordable5G architecture.
- Smartcity, which is a service for Computer Vision Analysis for Emergencies (CVAE)
- MCS, which is another service focused on Mission Critical Service over 5G.

First, section 2 defines the integration and testing methodology. In this section, we define the path to be followed in order to update the two initial platforms (Castellolí and Málaga) with all new building blocks developed by partners and to transform them into two full different affordable 5G solutions. The result will be different platforms as not all building blocks will be included in both platforms. Regarding the testing, in this section it is also explained the method to be followed, based on light test cases, and controlled by a test sequencing engine. With this approach, we can check the functionality of the building blocks and both final systems. In addition, some target KPIs can be measured.

Once the methodology is clear, in section 3 all new building blocks mentioned above are presented. These are the components that have been developed by partners and that will be integrated in the two platforms. A brief overview of each of them is included and, for some of them, also a table including their interfaces and how to access them.

In section 4 (Castellolí) and section 5 (Málaga), the specific integration and testing plans for each platform are presented. As mentioned, the final architectures of each platform will be different. Thus, a mapping of the architecture is included in each section, specifying the new building blocks that will be integrated in such location and also mentioning existing building blocks that were already in the platforms and that will be used in this project. These sections will therefore include a brief description of the already existing building blocks, which are not developed in this project, but which are essential for the complete operation of the whole system.

Finally, in section 6 some test cases to be performed in the platforms are presented. These test cases are separated into specific test cases that will be carried out just in one location (Castellolí or Málaga) and common test cases for both platforms. Due to the differences between each platform, there would be test cases that will be performed just in one of the platforms. On the other hand, common test cases for both platforms are also defined, which are all test cases that are identical for Castellolí and Málaga, as the same testing is necessary to be replicated in each location.

2 AFFORDABLE5G APPROACH TO THE INTEGRATION AND TESTING

2.1 Integration methodology

The integration methodology is based on the high-level architecture depicted in Figure 1, which can be differentiated into 3 main layers: The Management, Orchestration & Automation layer, the Network Function layer and the Infrastructure layer. The layer located at the top includes the elements capable of organizing the entire system. It includes ML elements, management elements and the orchestrator. The Network Function layer is mainly composed of the O-RAN element and the 5G core VNFs/CNFs. The O-RAN element is composed of O-RAN PNFs connected to O-RAN CNFs via Open Fronthaul. This represents the radio part and also includes near-RT RIC to enable near-real-time control and optimization of the elements and resources of this block. The 5G core VNFs/CNFs element is a Stand Alone core network that combines the necessary NFs for the user plane (UPF) and the control plane (AMF, SMF, PCF, UDM, NSSF). The 5G core also includes the NWDAF, which is responsible for providing network analysis information upon request from NFs. Finally, the Infrastructure Layer consists of the Core and Edge/Regional NFVIs, which are the cloud infrastructures where software is decoupled from hardware; the cell site platform, which includes proprietary infrastructures and where the software is coupled with the hardware; and transport network segments and synchronization elements.

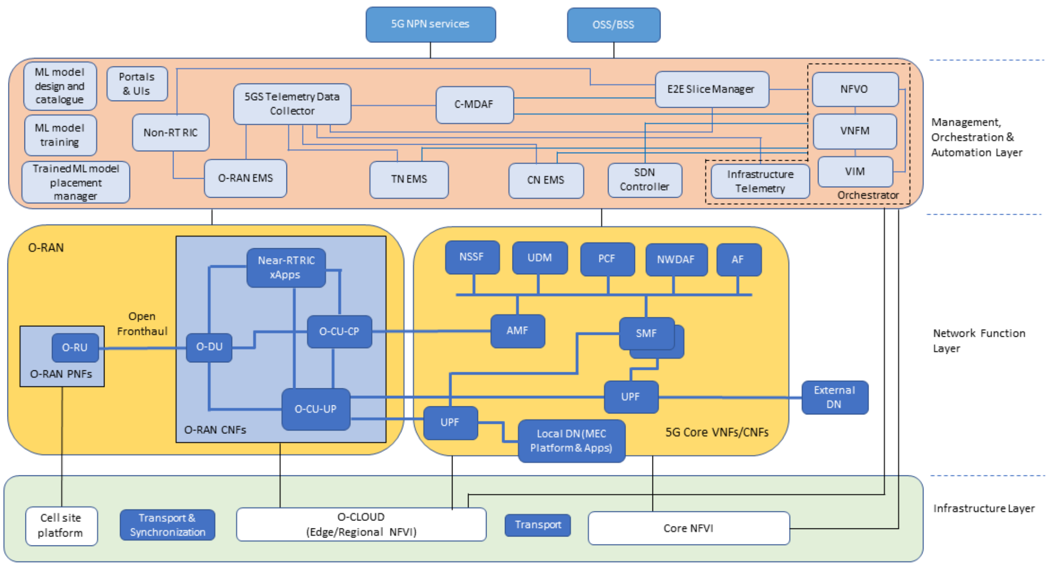


Figure 1: Affordable5G high level architecture

Based on the current state of Castellolí and Málaga platforms, the necessary elements will be replaced so that these platforms are updated and represent an architecture like that of Figure 1. The architecture of each of the platforms will not be exactly that of Figure 1, since not all components developed in this project will be integrated into both platforms and, therefore, the final architectures will be different in each location. Thus, it will be necessary to carry out a mapping for Castellolí platform (section 4) and another one for Málaga platform (section 5). In this mapping, the blocks that were initially on the platforms and the new blocks developed in Affordable5G will be described and differentiated.

Once the final mappings of each platform have been defined, testing will be carried out on both of them. The testing methodology used is based on the test sequencing engine OpenTAP [8] and the plugins developed in 5GENESIS project. This is described in detail in section 2.2.

The idea is to make use of this methodology to be able to include one by one the new building blocks that partners are developing, starting from the initial platforms of Castellolí and Málaga. In this way, we mitigate the risk of including new components in the platforms and we can carry out several integration phases, starting with tests in which only one or two new building blocks are involved and ending with a final test in which all new buildings blocks of the platform are included. Therefore, the final test is an E2E test of the complete architecture of each of the locations. It is important to note that in this section, one building block can be the result of the pre-integration of different minor building blocks in WP2 and WP3 (e.g., O-RAN).

For this methodology to work, it is essential that all those involved in the development of the new building blocks indicate whether these can be controlled, configured and / or monitored using OpenTAP. Therefore, in section 3, in the description of each building block (just in those which are accessible to OpenTAP) a table is included with the information necessary to know how OpenTAP can communicate with the block itself.

2.2 Testing methodology

The idea is to define and implement a light test cases-based method for testing integration of building blocks, considering the platforms operators as users.

2.2.1 Overview of the test cases-based testing

The methodology to be followed for the definition of the test cases in the Affordable5G project is based on the methodology defined as part of the 5Genesis project [9]. The full definition of this methodology can be found in Deliverable 2.4 of this project; however, a small description of this methodology can be found below.

In order to improve the repeatability and comparison of the results obtained during an experiment execution, all the details of the experiment are first specified as a Test Case. A Test Case includes information about the scope of the experiment, the target measurements and methods for the calculation of the KPIs, the pre-conditions for the experiment execution and the test sequence to follow, among others. The structure of a Test Case can be seen in Table 1, including a description of all the fields contained.

Table 1 : Test case template

Test Case Template		***-ID number-***	***-Related Metric ID-***
#	Description of the fields to be completed		
1	Description of the target KPI *Here goes the definition of the target KPI. Each test case targets only one KPI (main KPI). However, secondary measurements from complementary KPIs can be added as well (see field 4 in this template). The definition of the main KPI specializes the related target metric (the ID of the related target metric is declared in the first row of this template). More precisely, the definition of the main KPI declares at least the reference points from which the measurement(s) will be performed, the underlay system, the reference protocol stack level etc.*		

Test Case Template		-ID number-	-Related Metric ID-
2	Methodology *Here the acceptable values for the monitoring time, the iterations required, the monitoring frequency, etc., are declared. The reference to the calibration test is taken from the test case. This is to facilitate the comparison between measurements.*		
3	*Calculation process and output* *Here goes information related to the calculation process required. This information may include details related to the underlay system. Here goes also the Units of the metric, and potentially a request for first order statistics (Min, Max, etc.)*		
4	Complementary measurements *A secondary list of KPIs useful to interpret the values of the target KPI. Getting these measurements is not mandatory for the test case.*		
5	Pre-conditions *Any requirement that needs to be fulfilled before execution of this test case. A list of test specific pre-conditions that need to be met by the System Under Test (SUT) including information about **equipment configuration**, **traffic descriptor** i.e., precise description of the initial state of the SUT required to start executing the test sequence*		
6	Applicability *A list of features and capabilities which are required to be supported by the SUT in order to execute this test (e.g., if this list contains an optional feature to be supported, then the test is optional)*		
7	Test Case Sequence *Specializes the measurement process (methodology) of the metric for the selected underlay system. Measurements points and measurement procedure specification.*		

Once a Test Case has been defined, the specific implementation of the Test Case considers both the requirements described in the Test Case and the specific capabilities of the testbed where the experiment execution will take place, which means that a Test Case may be implemented differently depending on the environment.

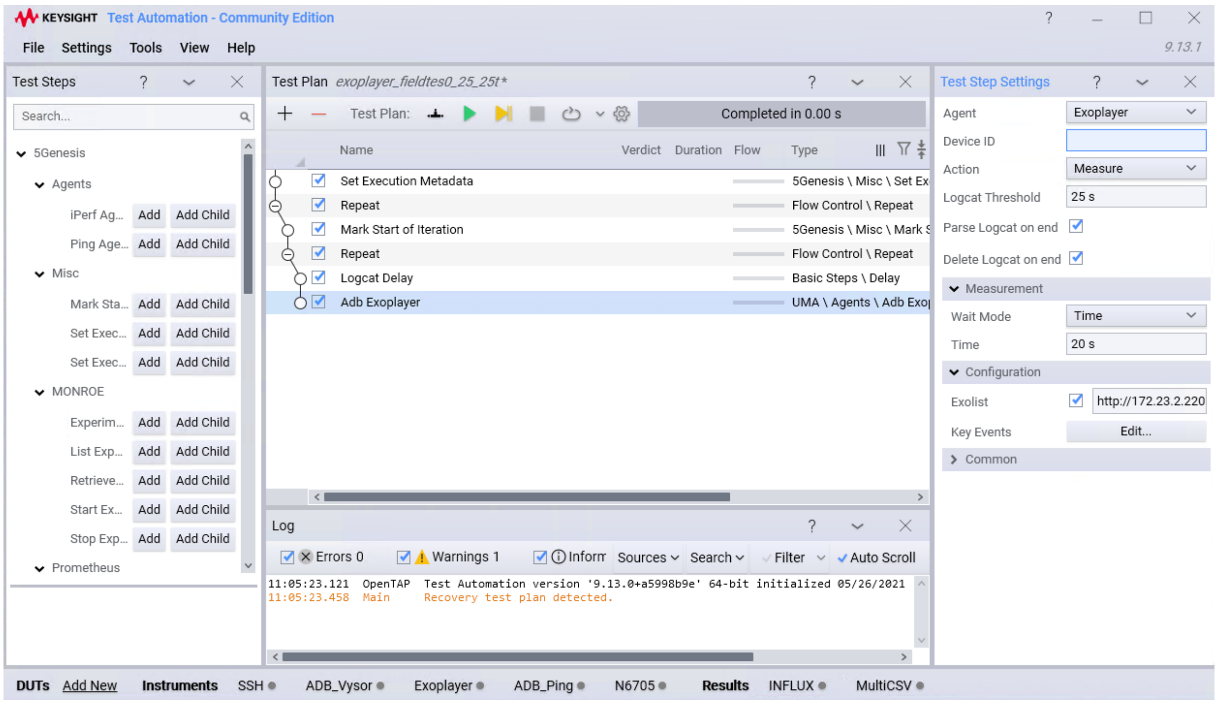
However, as long as the requirements of the Test Case are taken into account during the implementation process, the repeatability of the experiment will be possible to compare the results obtained, regardless of the platform where the test has been executed.

2.2.2 OpenTAP

OpenTAP is a test sequencing engine that provides functionality for the definition of the test logic, the management of heterogeneous devices (configuration, control and measurement extraction) and the handling of results within a single application.

In order to support a wide variety of components (both hardware and software based), OpenTAP is highly extensible: by leveraging the use of *.Net* libraries, end users are free to create new *TapPlugins*, which define new functionalities that are integrated directly in the OpenTAP interface. In addition, OpenTAP includes a set of plugins that provide support for common requirements out of the box.

Figure 2 shows the OpenTap interface with a brief example of a Test Plan.


KEYSIGHT Test Automation - Community Edition
File Settings Tools View Help
9.13.1
Test Steps
Search...
5Genesis
Agents
iPerf Ag... Add Add Child
Ping Age... Add Add Child
Misc
Mark Sta... Add Add Child
Set Exec... Add Add Child
Set Exec... Add Add Child
MONROE
Experim... Add Add Child
List Exp... Add Add Child
Retrieve... Add Add Child
Start Ex... Add Add Child
Stop Exp... Add Add Child
Prometheus
Test Plan exoplayer_fieldtes0_25_25t*
Test Plan:
Completed in 0.00 s
Test Step Settings
Log
Errors 0
Warnings 1
Inform
Sources
Search
Filter
Auto Scroll
DUTs Add New
Instruments SSH ADB_Vysor Exoplayer ADB_Ping N6705
Results INFLUX MultiCSV

Name	Verdict	Duration	Flow	Type
Set Execution Metadata				5Genesis \ Misc \ Set Ex
Repeat				Flow Control \ Repeat
Mark Start of Iteration				5Genesis \ Misc \ Mark S
Repeat				Flow Control \ Repeat
Logcat Delay				Basic Steps \ Delay
Adb Exoplayer				UMA \ Agents \ Adb Exo

Agent	Exoplayer
Device ID	
Action	Measure
Logcat Threshold	25 s
Parse Logcat on end	✓
Delete Logcat on end	✓
Measurement	
Wait Mode	Time
Time	20 s
Configuration	
Exolist	✓ http://172.23.2.220
Key Events	Edit...
Common	

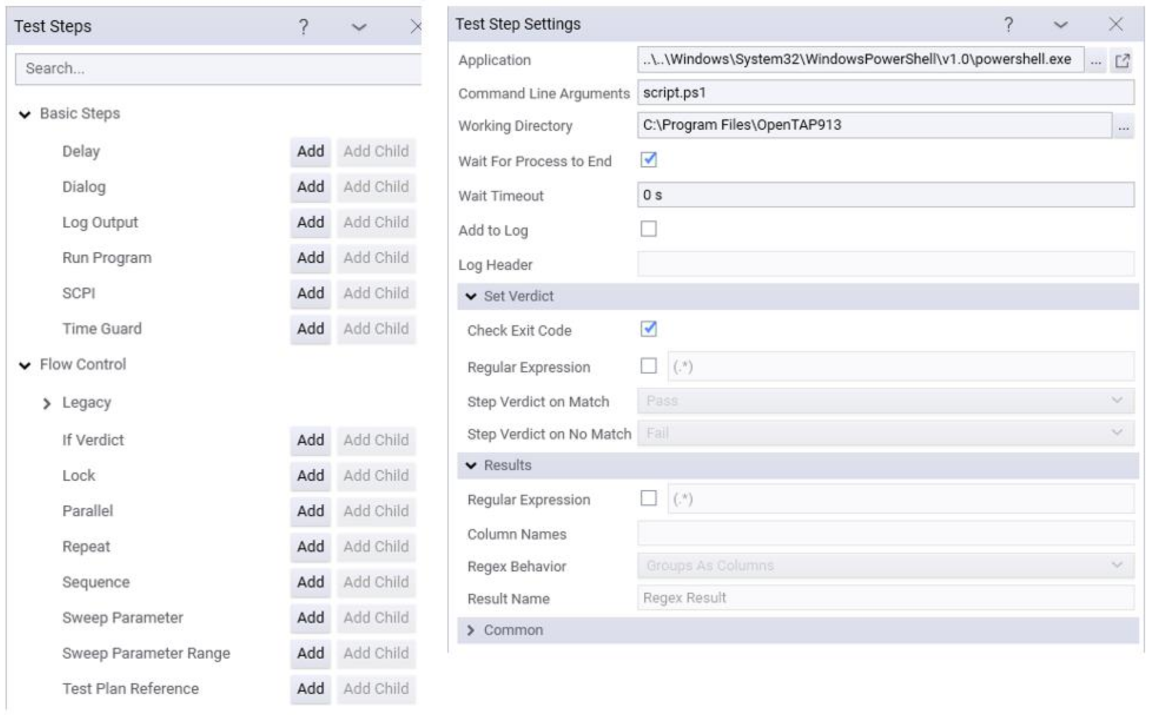
```
11:05:23.121  OpenTAP  Test Automation version '9.13.0+a5998b9e' 64-bit initialized 05/26/2021
11:05:23.458  Main     Recovery test plan detected.
```

Figure 2: OpenTAP interface

Test cases are implemented in the form of *TAP TestPlans*. A Test plan is a collection of *Test Steps* (which in turn define each basic action that is performed during a *TestPlan* execution), arranged in a tree structure. This structure, along with the usage of specific *Test Steps* define the order and logic implemented during a *TestPlan* execution, much like the flow control primitives used in programming languages.

As mentioned, *Test Steps* are the basic building blocks of a *TestPlan*. Each step defines a single action that can take place during a testplan execution. However, these actions can be as complex as necessary; for example, a kind of *Test Step* can be used for adding a single message to the execution log, other kind can be used to specify the repetition of certain child steps, and a more complex one can be used to control the complete measurement cycle of a particular instrument.

Figure 3 displays the options to configure and sketch out a Test Plan.



Test Steps

Search...

- Basic Steps

Delay	Add	Add Child
Dialog	Add	Add Child
Log Output	Add	Add Child
Run Program	Add	Add Child
SCPI	Add	Add Child
Time Guard	Add	Add Child

- Flow Control
  - Legacy

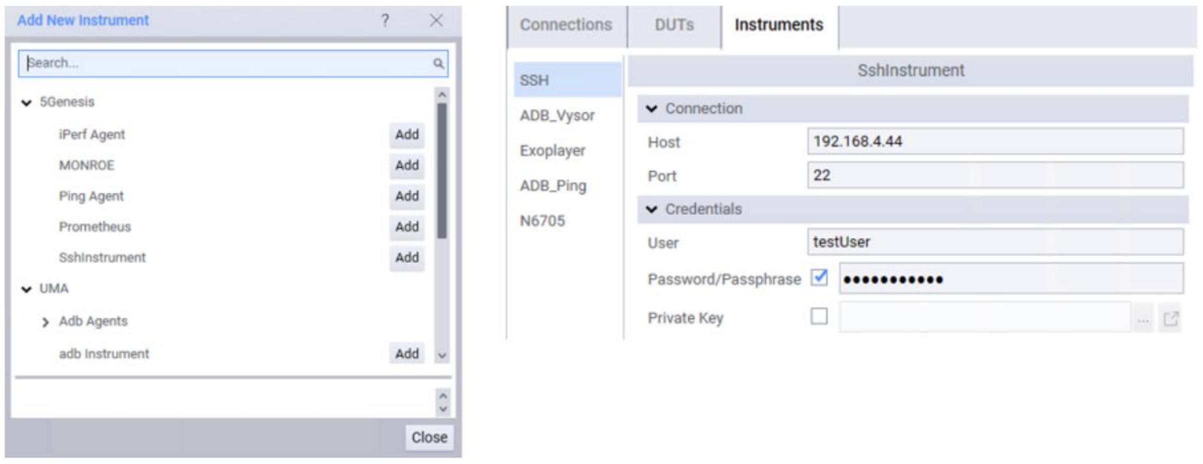
If Verdict	Add	Add Child
Lock	Add	Add Child
Parallel	Add	Add Child
Repeat	Add	Add Child
Sequence	Add	Add Child
Sweep Parameter	Add	Add Child
Sweep Parameter Range	Add	Add Child
Test Plan Reference	Add	Add Child

Test Step Settings

Application	..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Command Line Arguments	script.ps1
Working Directory	C:\Program Files\OpenTAP913
Wait For Process to End	☑
Wait Timeout	0 s
Add to Log	☐
Log Header	
Set Verdict	
Check Exit Code	☑
Regular Expression	☐ (.*)
Step Verdict on Match	Pass
Step Verdict on No Match	Fail
Results	
Regular Expression	☐ (.*)
Column Names	
Regex Behavior	Groups As Columns
Result Name	Regex Result
Common	

Figure 3: OpenTAP Test Step Settings

Normally, all *Test Steps* contain a set of *Test Step Settings*, which are used for specifying in more detail the behaviour of a particular step, in comparison with other steps of the same kind. In order to abstract the specific details and functionality used for the management of a particular component, OpenTAP introduces the concepts of *Instrument* (Figure 4) and *DUT* (Device Under Test). Though functionally equivalents, *Instruments* are normally used to encapsulate and abstract the use of measurement equipment and other testbed devices, while the *DUT* concept is reserved for the kind of devices that are subject to the testing procedure.



Add New Instrument

Search...

- 5Genesis

iPerf Agent	Add
MONROE	Add
Ping Agent	Add
Prometheus	Add
SshInstrument	Add

- UMA
  - Adb Agents

adb Instrument	Add

Close

Connections | DUTs | **Instruments**

SSH, ADB_Vysor, Exoplayer, ADB_Ping, N6705

SshInstrument

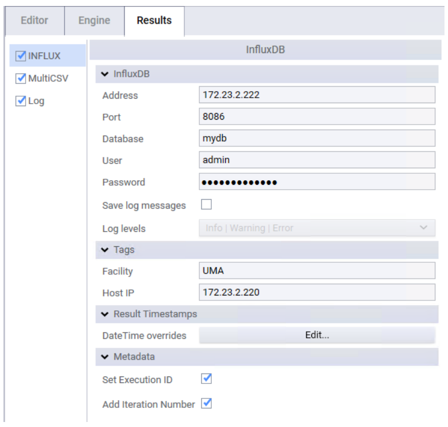
Connection	
Host	192.168.4.44
Port	22
Credentials	
User	testUser
Password/Passphrase	☑ ●●●●●●●●●●●
Private Key	☐

Figure 4: OpenTAP Instruments interface

For example, using this (optional) nomenclature, a *DUT* may be an Android device that is being profiled as part of the test, and *Instruments* may be the power analyser used to measure battery consumption and a remote server controlled through SSH that is sending data to the device via the radio interface.

Finally, in order to handle the storage of results, OpenTAP introduces the concept of *ResultListener*. In OpenTAP, results generation is decoupled from the actual storage of the measurements, which improves extensibility and eases the management of both processes:

- Results are generated by *Test Steps*. Each step may generate results as needed, which are then *Published* in the form of *ResultTables*. These tables may contain any number of rows and columns.
- The published *ResultTables* are retrieved by any number of enabled *ResultListeners* and are handled by each one of them independently.



✓ INFLUX	
✓ MultiCSV	
✓ Log	

InfluxDB

InfluxDB

Address	172.23.2.222
Port	8086
Database	mydb
User	admin
Password	•••••••••••••
Save log messages	☐
Log levels	Info | Warning | Error

Tags

Facility	UMA
Host IP	172.23.2.220

Result Timestamps

DateTime overrides	Edit...

Metadata

Set Execution ID	✓
Add Iteration Number	✓

Figure 5: OpenTAP Results configuration

By using this architecture, end users are able to configure and use as many different storage solutions as needed, encapsulating the specific details of each solution to a separate *ResultListener*. For example, Figure 5 shows a setup where all log messages are saved to a text file, while results are recorded at the same time in an InfluxDb database and as multiple CSV files in the hard disk.

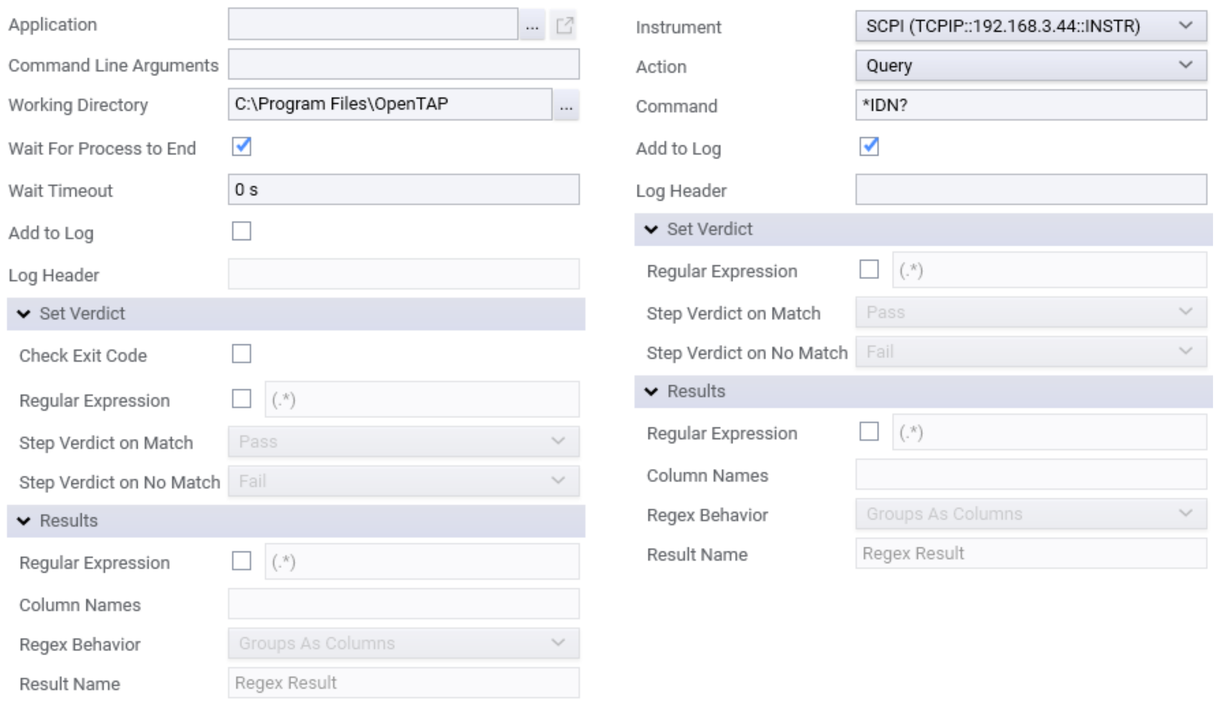
2.2.3 Implementation aspects for Affordable5G

In order to orchestrate all components during the execution of the test cases, each component that takes part of the experiment must be able to provide some sort of control and/or configuration interface. For components that are able to generate results, it must be possible to retrieve these measurements through this interface in order to be possibly automated.

For those cases where it is not desirable to develop a TAP plugin for controlling the device, OpenTAP provides some generic Test Steps that can be used for accessing certain interfaces, while others can be controlled using open-source TAP plugins. This section provides an overview of some of the possible solutions that can be adopted when integrating heterogeneous equipment for the execution of a test case.

Out of the box, OpenTAP includes two Test Steps that allow the control of devices in the most generic way. These are the “Run Program” and “SCPI” steps.

The **Run Program** steps allows the execution of arbitrary scripts and applications through the command line interface of the machine where OpenTAP is deployed. This provides access to a wide range of possibilities, which includes the usage of utilities for the configuration and control of the device that are included by the equipment vendor, or generic tools like *curl*, for controlling REST interfaces. It is also possible to use this step for executing custom made scripts that are prepared specifically for the device to control. This step can also be configured for using regular expressions, that can extract results from the output of the command execution. Figure 6 (left) shows the available settings for this step.



Application	
Command Line Arguments	
Working Directory	C:\Program Files\OpenTAP
Wait For Process to End	✓
Wait Timeout	0 s
Add to Log	☐
Log Header	
Set Verdict	
Check Exit Code	☐
Regular Expression	☐ (.*)
Step Verdict on Match	Pass
Step Verdict on No Match	Fail
Results	
Regular Expression	☐ (.*)
Column Names	
Regex Behavior	Groups As Columns
Result Name	Regex Result

Instrument	SCPI (TCPIP::192.168.3.44::INSTR)
Action	Query
Command	*IDN?
Add to Log	✓
Log Header	
Set Verdict	
Regular Expression	☐ (.*)
Step Verdict on Match	Pass
Step Verdict on No Match	Fail
Results	
Regular Expression	☐ (.*)
Column Names	
Regex Behavior	Groups As Columns
Result Name	Regex Result

Figure 6: OpenTAP « Run Program » configuration

The **SCPI** (Standard Commands for Programmable Instruments) step can be used for controlling generic SCPI instruments. As is the case for the Run Program step, basic result generation can be achieved by using regular expressions. The settings for this step can be seen in Figure 6 (right).

The following solutions are not included by default in OpenTAP, however, they are free for use and available as open source software. The main advantage in this case is that, even though a similar functionality could be achieved by using the Run Command step in conjunction with the required tools, these TAP plugins provide a more fine-tuned user interface for the specific details of the underlying interface that they expose.

For equipment and devices that can be managed through the SSH interface, it is possible to make use of the **UMA SSH TAP Plugin** [10]. This plugin provides access to a set of test steps that allow the execution of arbitrary commands in the remote device, as well as for sending files and directories between the OpenTAP machine and the remote equipment. This TAP Plugin provides an easy way for integrating such devices: for example, different scripts for performing certain actions can be started remotely by OpenTAP, while any generated measurement can be saved on a file in the remote machine that can later be retrieved for further processing.

In order to automate the usage of Android devices, the **UMA Android TAP plugin** [11] provides access to common ADB (Android Debug Bridge) use cases, such as transferring files from and to the device, installing and starting applications, etc, in an easy-to-use interface. Additionally, the provided steps allow the execution of arbitrary commands in the device: both pure ADB commands and shell commands that can be sent to the underlying OS of the Android device.

For applications that are specifically prepared for automation, this TAP plugin can be used for sending the required *intents* for each of the required actions, while other applications can be automated by simulating the usage of the touch screen or the keyboard via ADB commands. For those cases where the equipment cannot be controlled by an interface that is already covered by an existing TAP plugin, or in cases where more fine-tuned functionality is required, it is always possible to extend or develop a TAP plugin for that particular component.

If deemed necessary or if no compatible interface is available, it is possible to create new TapPlugins specifically for certain components. The usage of the OpenTAP SDK gives access to any of the available .Net libraries that may be required in order to have access to the required interfaces for the component. The SDK also allows the integration within the OpenTAP environment, giving access to the generic functionality required for the development of Test Steps or Result Listeners, as well as several helpers and pre-implemented functions.

2.2.4 Test plans examples

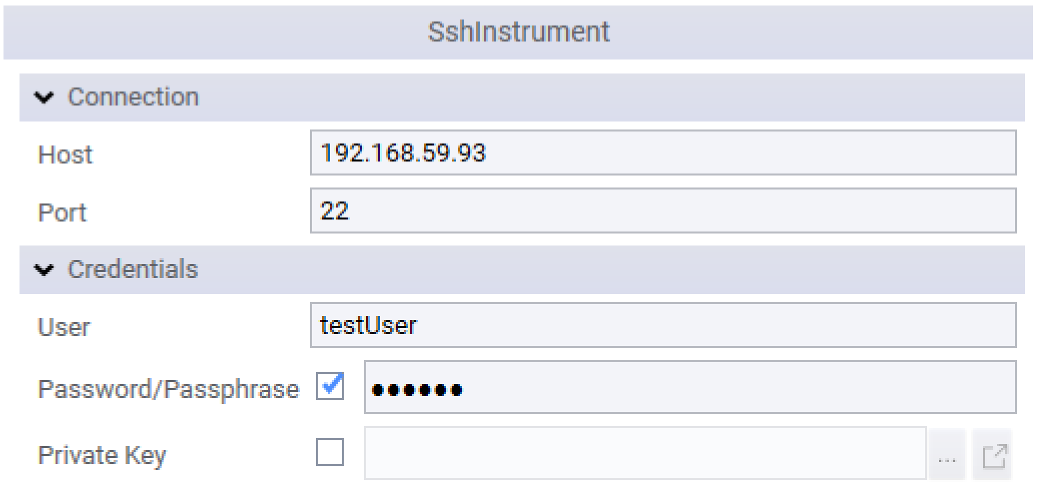
2.2.4.1 Basic Latency test using the SSH interface

The following example shows the implementation of a simple latency test using OpenTAP, while also showing how to integrate equipment in the testbed quickly by using readily available plugins. The test case sequence consists of:

1. Remotely starting the execution of ping in one node of the testbed.
2. Recording the output generated in a text file.
3. Retrieving the text file from the remote node, for storage and further (offline) processing.

In order to implement this test case, the open source SSH TAP Plugin [10] will be used in order to simplify the definition of the testplan, however, it should be noted that the same ideas may be implemented by using other SSH implementations available in the same environment as OpenTAP, by making use of the “Run Program” test step that is included out of the box.

The first step for the implementation of the test case is to define the connection details of the machine that will be used as remote node, as well as specifying the credentials of the user that will run the commands in the machine. This is done by creating a new SSH Instrument in the OpenTAP interface as can see in Figure 7.


SshInstrument

Connection

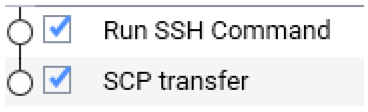
Host	192.168.59.93
Port	22

Credentials

User		testUser
Password/Passphrase	✓	••••••
Private Key	☐	

Figure 7: Creation of a SSH Instrument on OpenTAP

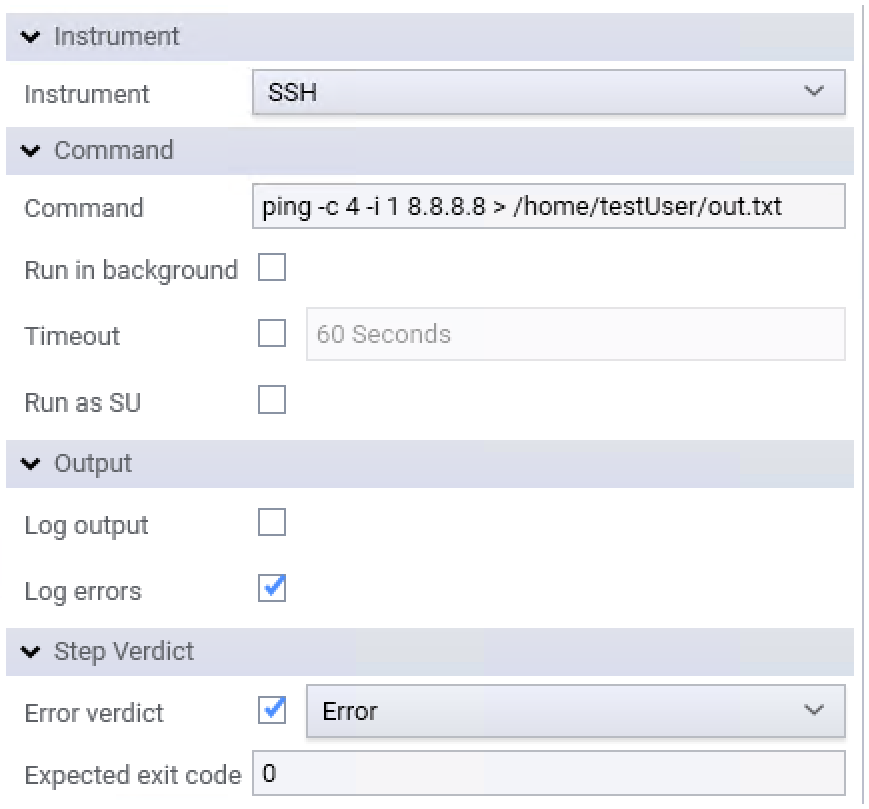
The actual test plan is composed by two steps (Figure 8), the first one performs the execution of the ping test, while the second retrieves the generated file from the remote machine.


✓ Run SSH Command

✓ SCP transfer

Figure 8: Test Plan Steps

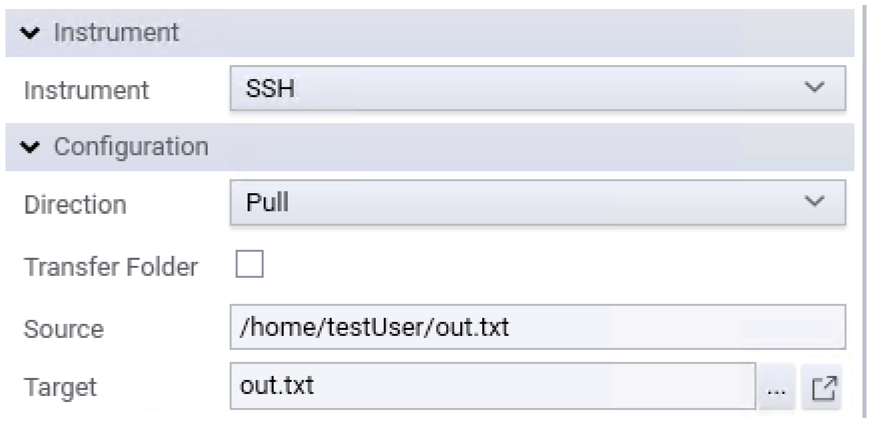
The configuration of the “Run SSH Command” step can be seen in Figure 9. The step setting defines the execution of a ping command that sends 4 ICMP requests to a known IP address, with an interval of 1 second. By using the redirection capabilities of the remote terminal, the output of this command is sent to a text file in the home directory of the user.



Instrument	
Instrument	SSH
Command	
Command	ping -c 4 -i 1 8.8.8.8 > /home/testUser/out.txt
Run in background	☐
Timeout	☐ 60 Seconds
Run as SU	☐
Output	
Log output	☐
Log errors	✓
Step Verdict	
Error verdict	✓ Error
Expected exit code	0

Figure 9: Configuration of Run SSH Command

The second step then retrieves the file from the remote machine, as specified by the settings shown in Figure 10.



Instrument	
Instrument	SSH
Configuration	
Direction	Pull
Transfer Folder	☐
Source	/home/testUser/out.txt
Target	out.txt

Figure 10: Settings on OpenTAP to retrieve a file from a remote machine

This example does not detail the generation of results and considers that any required processing of the measurements will be performed outside of OpenTAP. However, if such functionality is needed, some possible approaches are mentioned below:

- If further processing is needed but it is not necessary to integrate the results generation with OpenTAP, it is possible to automatically request the execution of a specialized parsing script after the file has been received.
- Similar to the previous approach, but in order to integrate the results with OpenTAP, a Test Step may be developed, which calls the specialized script, reads the results, and publishes them in a compatible format.

- The post-processing of the file may be done inside OpenTAP, as part of the developed Test Step, instead of calling to a separate script.
- The SSH TAP may be used as a base for this functionality, integrating all the steps in the example testplan into a single step. This can be done either by editing the code, or by referencing the pre-built DLL of the SSH TapPlugin.

2.2.4.2 Throughput test using specialized plugins

This example provides a more in-depth look at the creation of more complex test plans, also providing insight about the development of specialized TAP plugins for specific components. The test consists of a throughput measurement between a remotely controlled PC and an Android device using iPerf [12] probes. The test sequence is:

1. An iPerf server instance is started in the remote PC
2. The iPerf client is started on the Android device, connecting to the iPerf server

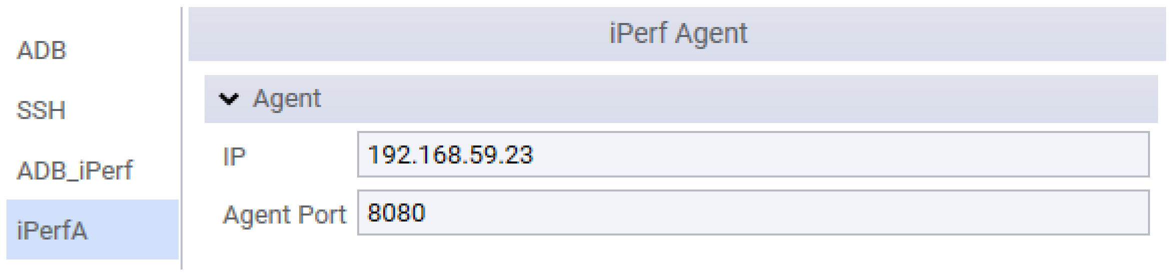
In this example we make use of several open-source TAP plugins and associated probes to implement the test case, which are briefly described below.

The automation of the PC version of iPerf is handled by the 5Genesis Remote iPerf Agent, which is composed by two elements, the Remote iPerf Agent itself [13], and the control plugin, which is included as part of the 5Genesis TAP plugins package [14].

The Remote iPerf Agent provides a REST interface that can be used for controlling the actual iPerf executable running on the machine, and also provides endpoints for retrieving the generated results in JSON format. The TAP plugin makes use of this interface for configuring and starting the iPerf probes, and then retrieves the results, which are then published in a compatible way for the available Result Listener.

In a similar way, on the Android device automation is also handled by two separate components, however, the interface used in this case is ADB. The iPerf instance is wrapped in this case by the UMA Android iPerf Agent [15] and the UMA ADB Agents package [16] which extends the functionality provided by the base UMA Android package [11]. Both provide capabilities similar to those mentioned for the PC alternative.

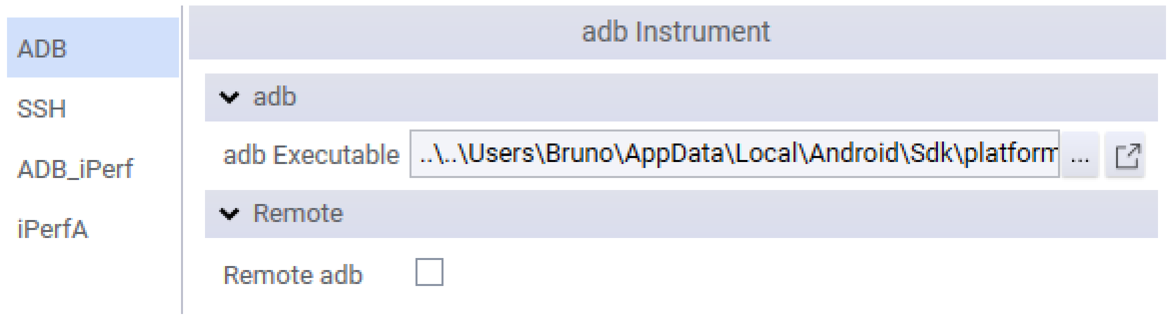
As with the previous example, the implementation of the test case starts with the definition of the Instruments that take part in the execution. In this case we need to define three instruments. The Remote iPerf Agent instrument contains the address and port where the provided REST interface is listening (Figure 11).



IP: 192.168.59.23
Agent Port: 8080

Figure 11: Remote iPerf Agent

The ADB agent is configured by two separate instruments. The first one (Figure 12) specifies the location of the ADB executable in a generic form, which can then be used by multiple agents at the same time. The second instrument (Figure 13) only needs to include a reference to the generic ADB instrument defined first.


ADB
SSH
ADB_iPerf
iPerfA

adb Instrument

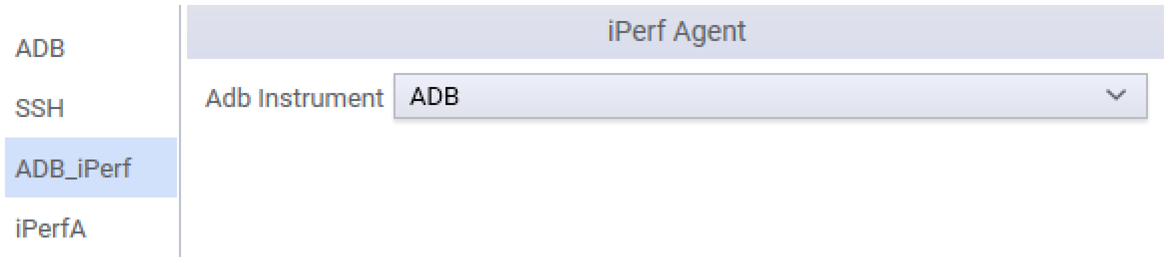
adb

adb Executable: ..\..\Users\Bruno\AppData\Local\Android\Sdk\platform

Remote

Remote adb: ☐

Figure 12: First instrument needed to configure adb agent


ADB
SSH
ADB_iPerf
iPerfA

iPerf Agent

Adb Instrument: ADB

Figure 13: Second instrument needed to configure adb agent

This test case can also be defined by using two test steps, however, in this case these are arranged in a parent-child relation, since the duration of the execution of the parent step will depend on the duration of the child step, as we will detail below. The test plan can be seen in Figure 14.

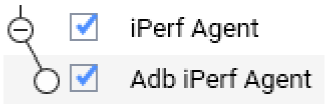
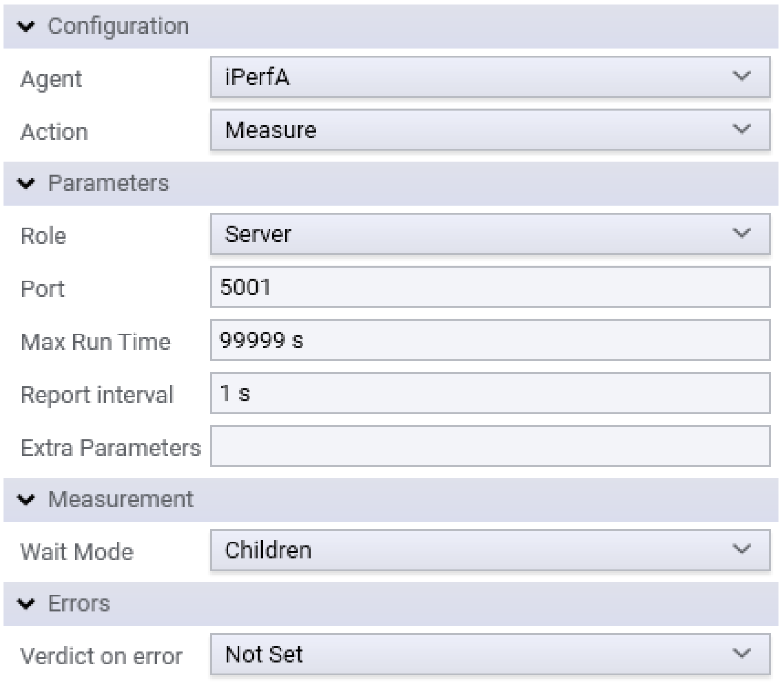

☑ iPerf Agent
☑ Adb iPerf Agent

Figure 14: Test plan to configure iPerf server and ADB agent

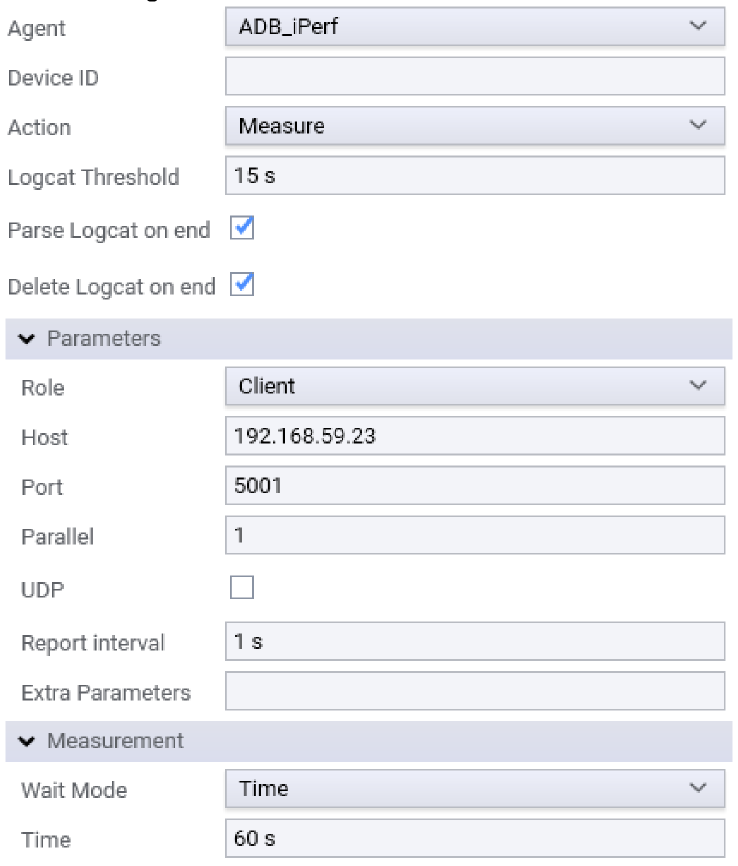
The first step configures the iPerf server instance that runs in the remote PC. Settings of note include the “Measure” action, which is used in order to perform all the measurement cycle (start, stop and measurement collection) in a single step (if necessary, the step can be configured for performing only one of these actions, in case the plugin needs to be used in a more complex test plan with multiple components). The measurement mode is set to “Children”, which specifies that the duration of the measurement will be controlled by the execution of the child steps. Figure 15 shows all the settings used in this step.



Configuration	
Agent	iPerfA
Action	Measure
Parameters	
Role	Server
Port	5001
Max Run Time	99999 s
Report interval	1 s
Extra Parameters	
Measurement	
Wait Mode	Children
Errors	
Verdict on error	Not Set

Figure 15: Settings to configure Measure action

The child step configures the client instance that runs on the Android device. This step is very similar to the previous one but includes some extra settings for controlling the usage of the adb interface. The wait mode in this step is set to "Time", with a duration of 60 seconds. The settings on this step can be seen in Figure 16.



Agent	ADB_iPerf
Device ID	
Action	Measure
Logcat Threshold	15 s
Parse Logcat on end	✓
Delete Logcat on end	✓
Parameters	
Role	Client
Host	192.168.59.23
Port	5001
Parallel	1
UDP	☐
Report interval	1 s
Extra Parameters	
Measurement	
Wait Mode	Time
Time	60 s

Figure 16: Settings to configure Measure action with extra options

3 BUILDING BLOCKS

This section describes all the building blocks belonging to the integration architecture, and also services that will be running over this architecture, following an established format to obtain all the necessary information. In each of the building blocks, information is given about its content and operation, including figures to support it. If the building block allows it, the architecture is also included to represent how it is formed and how the integration with the whole system has been allowed. Finally, a table has been included with all the necessary information (interfaces, APIs, etc.) that could be needed to access each building block through OpenTAP.

3.1 Hardware acceleration

At the hardware level, one of the building blocks of the Affordable5G architecture is the NEOX™ [1] embedded multicore inference processor of Think Silicon. NEOX (Figure 17) is a parallel multicore and multithreaded GPU architecture based on the RISC-V RV64C ISA instruction set [2]. In the final version of the product, the number of cores will vary from 4 to 64 cores organized in 1-16 cluster elements with configurable cache sizes and thread counts. Depending on cluster / core configuration, NEOX compute power is expected to range from 12.8 to 409.6 GFLOPS at 800MHz with support for FP16, FP32 and optionally FP64 and SIMD instructions.

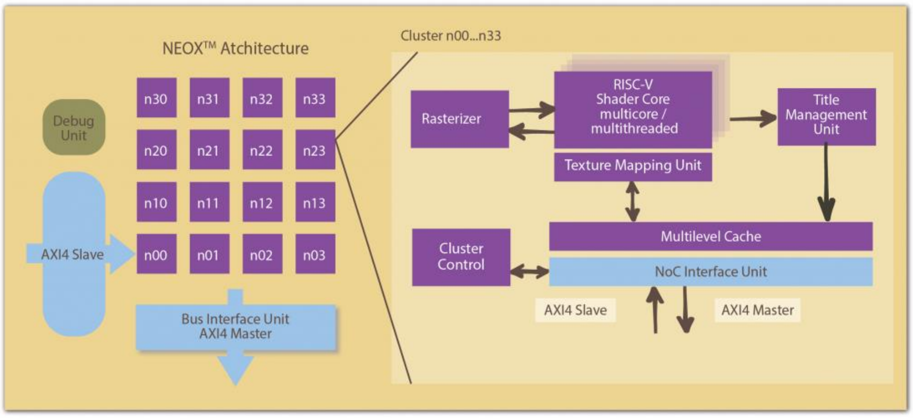


Figure 17: The NEOX architecture

The competitive advantage of the NEOX accelerator is its low power consumption, thus the target is to use the NEOX accelerator (realized in an FPGA board) for executing the AI/ML applications at the edge level. Currently, two specific applications have been identified: the Computer Vision Analytics for Emergencies (CVAE) applications (developed by UBI) and ML-based 5G power management applications at the edge level (developed by NKUA).

More specifically, for the CVAE applications, two main software modules will be offloaded in the NEOX accelerator: The Edge Object Detection and the Edge Object Classification. The NEOX accelerator will be integrated and prototyped in an FPGA device and connected to the rest of the system through a dual core ARM processor (running a standard Linux OS). The target will be to execute these modules with minimal energy consumption.

Apart from the hardware architecture, the NEOX processor is associated by an end-to-end SDK tool, called AI SDK. NEOX AI SDK is a collection of many open source and proprietary tools for analysing, visualizing, converting, compressing, and deploying on pre-trained and post-trained AI/ML/NN models on the NEOX architecture. The AI SDK is based on TensorFlow Lite [3] and it supports the most widely used model formats. The tools allow to perform various iterative steps in model compression and model analysis, until the desired balance between "accuracy-performance-memory" is achieved. More information about NEOX AI SDK can be found in D2.1 [4].

As noted, the NEOX accelerator will be realized in a high-capacity FPGA platform with the codename NEMA®|Bits [5] (mainly used as a pre-sales tool). NEMA®|Bits is an out-of-box software environment designed to showcase and validate the graphics and AI technology developed by Think Silicon. NEMA®|Bits acts as a vertical (including both hardware and software components) demonstration platform, where the interested users and customers can download the latest developments of the company in their Xilinx XC706 [6] Evaluation boards. To ease the installation process, a fully functional, ready-to-use SD card is provided.

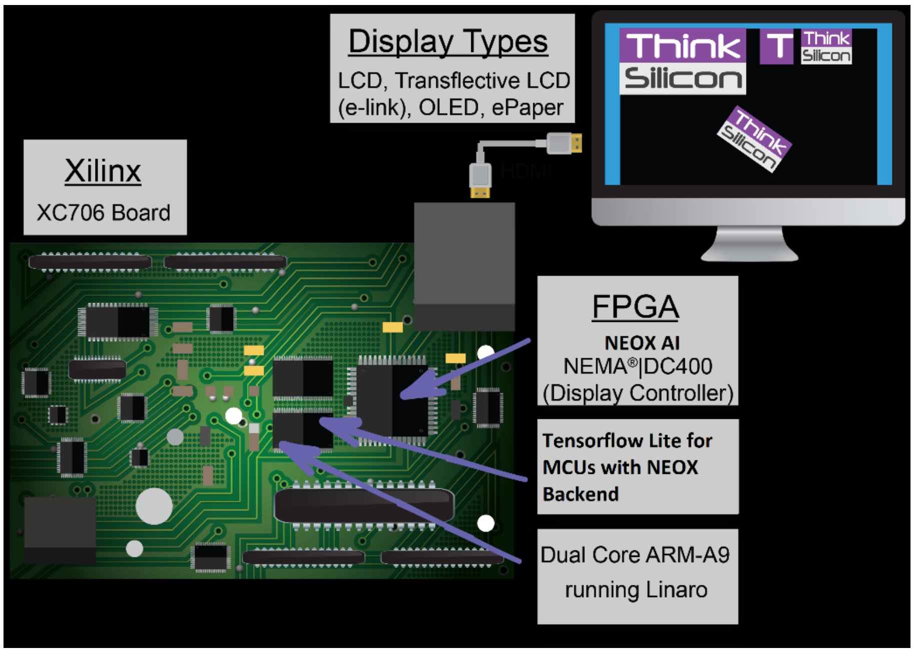

Display Types
LCD, Transflective LCD (e-link), OLED, ePaper
Think Silicon
Xilinx
XC706 Board
HDMI
FPGA
NEOX AI
NEMA®|DC400 (Display Controller)
Tensorflow Lite for MCUs with NEOX Backend
Dual Core ARM-A9 running Linaro

Figure 18: The NEOX|Bits prototyping platform

The current version of NEOX®|Bits contains two Hardware IP Cores: NEOX® AI-GPU and NEMA®|DC-400 display processor. The SD card of NEOX|Bits contains the IP Cores of the company, the Linaro (Linux) Operating System, device drivers, and the AI/ML run-time system (based on Tensorflow Lite for MCUs) along with specific examples. The Xilinx XC706 contains ethernet ports while the host CPU (ARM processor) runs an embedded version of the Ubuntu OS (called Linaro). Therefore, the communication of the NEOX accelerator with the rest of the Affordable5G project will be done through standard ssh and sftp commands.

3.2 5G Core

Athonet is providing Affordable5G with a SA fully virtualized 5GC, tailored to the project's requirements to serve at best the project's use cases. The softwarized architecture of Athonet's 5GC offers increased flexibility and adaptability. Its 3GPP-compliant NFs, fully developed and programmed in-house at Athonet, are implemented as containerized VNFs, with the separation of UP and CP as required by 3GPP. Figure 19 depicts the VNFs that already compose the 5GC or that are currently under development and will be made available to the project within a few months after the publication of this document. A detailed description of each of these VNFs and their functionalities is provided in D3.1 [7]. Among the numerous features of the 5GC, it is worth mentioning the possibility of distributing core NFs to the edge of the network to enable edge computing and the capability to handle network slicing to deploy isolated and independent services with performance guarantees.

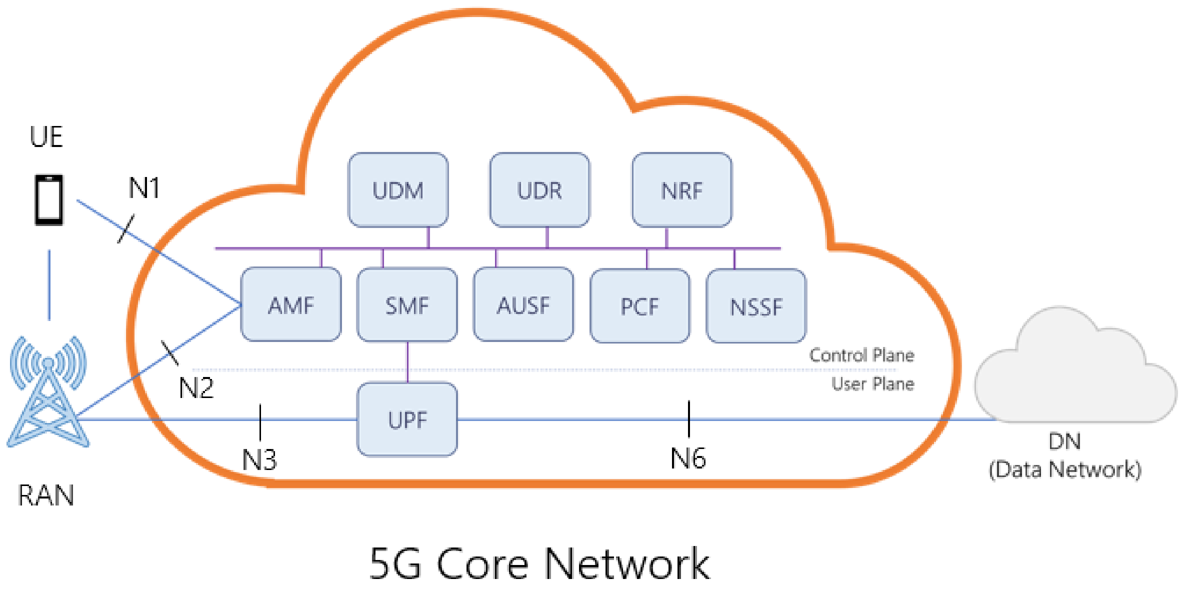


Figure 19: Athonet's 5G virtual core network functions

The following table summarizes the interfaces through which the 5GC and the UPF distributed at the edge of the network can be tested.

Table 2: 5GC and UPF interfaces

	5G Core	**UPF at the edge**
Control	N/A	N/A
Configuration	APIs (QoS available later on with device/apps adoption)	APIs
Monitoring	Prometheus	Prometheus

3.3 Affordable5G RAN

The Affordable5G RAN (Radio Access Network) is conformant to a 3GPP NG-RAN as described by [17]. Additionally, the Affordable5G RAN is also aligned to the O-RAN Alliance architecture [18], adding the Near-RT RIC (RAN Intelligent Controller) and associated interfaces (like O1/O2/A1/E2) as highlighted below in Figure 20.

The main components that, along with Near-RT RIC, compose the RAN architecture are: O-CU (Open Central Unit), O-DU (Open Distributed Unit) and O-RU (Open Remote Unit). All these elements are also described in detail below.

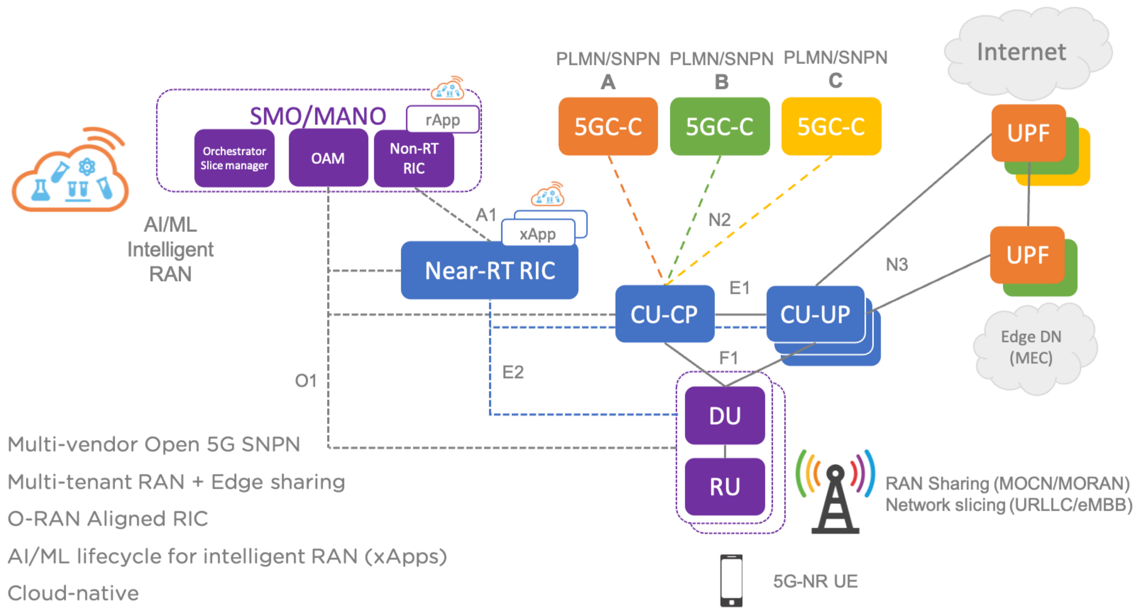


Figure 20: Affordable5G 3GPP NG-RAN and O-RAN Alliance aligned RAN

The Affordable5G RAN is further detailed in Affordable5G D3.1 [7].

3.3.1 O-CU

The CU, or Central Unit, implements the 'L3' part of the NG-RAN gNB. The CU interfaces to the gNB DU via the F1 mid-haul interface and the mobile core (5GC) via the N2 and N3 backhaul interfaces (NG-C/NG-U).

The CU is split between the control plane and user plane:

- CU-CP: Control-plane of CU, implementing the RRC (Radio Resource Control) functions as described in [17], chapter 7. Communicated to DU via F1-C and to the 5GC control-plane via N2.
- CU-UP: User plane function of CU, implementing the PDPC ([17] §6.4) and SDAP (mapping of PDU sessions and their QoS flows to the access stratum DRB Data Radio Bearers, [17] §6.5). See Figure 21.

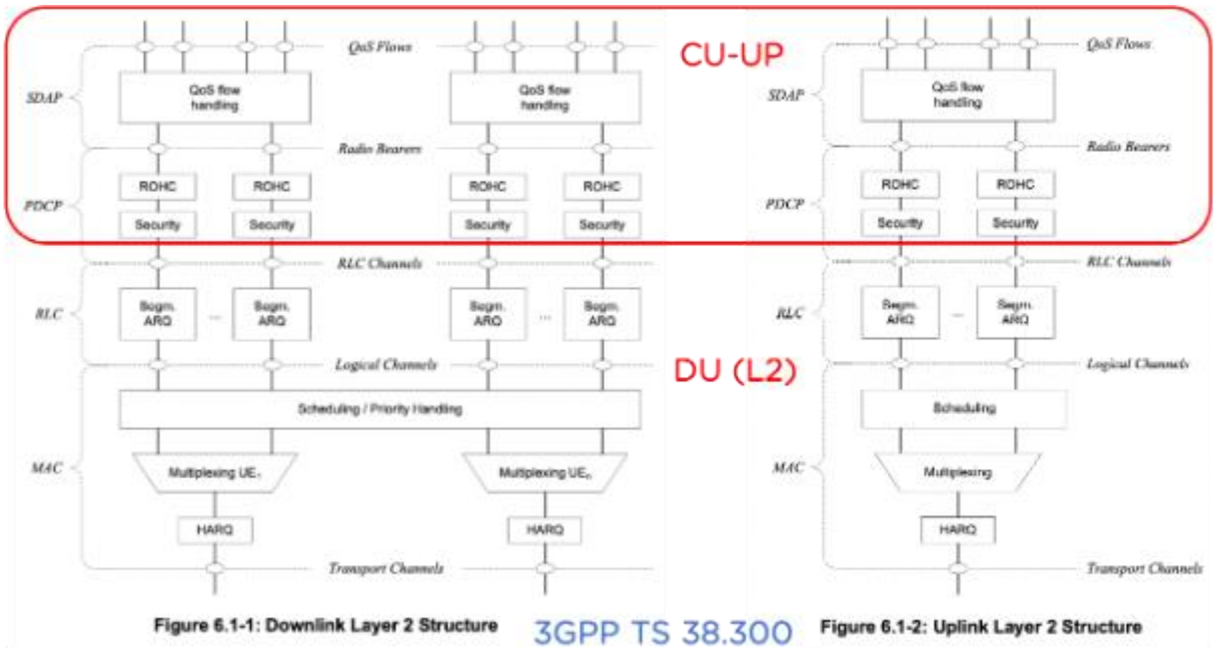


Figure 21: CU-UP functions, part of 3GPP TS 38.300 L2

The CU-CP controls the CU-UP via the E1 interface. The Affordable5G O-CU is further detailed in Affordable5G D3.1 §2.1.3.

Table 3: O-CU interfaces

O-CU	**O-RAN conformant SMO APIs**	**Alternative scripting capability**
Lifecycle/Orchestration	O-RAN O2 API [19]	Helm charts [21]
Configuration (FCAPS)	O-RAN O1 API [20] §2.1	Helm charts [21] (no dynamic reconfiguration, restart needed)
Monitoring	Via E2 to Near-RIC xApp telemetry exporter (Kafka/Prometheus), emulation of O1 ONAP VES	Rest APIs

3.3.2 O-DU

The overall of 5G NR entities is split into centralized unit (CU) and distributed unit (DU) by means of the F1 functional split, as defined in TS 38.401. DU is usually located in a range of several kilometres from the RU and runs the parts of the 3GPP PHY layer, MAC and RLC layers (see Figure 22). This logical node includes a subset of the gNB functions, depending on the functional split option, and its operation is controlled by the CU, and possibly by the RIC via the ORAN E2 interface. A single DU controls one or more RUs via a fronthaul interface (FHI) such as ORAN 7.2x functional split.

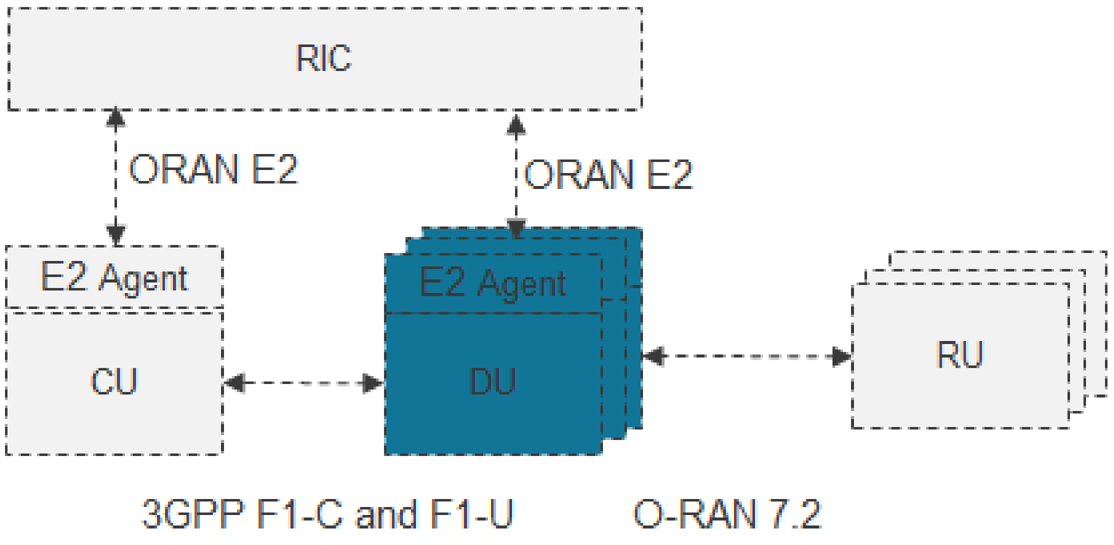


Figure 22: 5G NR functional split CU, DU, RU

The Affordable5G O-DU is further detailed in Affordable5G D3.1 §2.1.2.

Table 4: O-DU interfaces

O-DU	**O-RAN conformant SMO APIs**	**Alternative scripting capability**
Lifecycle/Orchestration	O-RAN O2 API [22]	Trirematics
Configuration (FCAPS)	O-RAN O1 API [20] §2.1 with specific DU O1 conformance to [23]	Defined layer-based service models as well as RestApi on the top of FlexRIC
Monitoring	Via E2 to Near-RIC xApp telemetry exporter (Kafka/Prometheus), emulation of O1 ONAP VES	Defined layer-based service models as well as RestApi on the top of FlexRIC

3.3.3 O-RU

The Radio Unit (RU) or ORAN Radio Unit O-RU is the unit at the edge of the Access Network. It interfaces with the O-DU northbound via O-RAN Interface and with the 5G User Terminals and IoT Sensors via air Interface as depicted in Figure 23 below.

The O-RU implements the Physical Layer (layer-1) split defined by the O-RAN association as option 7.2, therefore the O-RU device implements *Low-PHY* function that is connected to the *Hi-PHY* in O-DU.

RunEL selected the Sparq-2020-ORU [24] as the candidate to perform the RU functionalities needed in the Affordable 5G due to the following main reasons:

- Includes ORAN Interface between the O-RU and the O-DU, therefore the integration process is expected to be simple and fast.
- Has an embedded 3.3 to 3.8 GHz beam forming antenna that is within the 5G spectrum bands in Europe.
- It is programmable, based on FPGA, so it will be relatively simple to make changes if necessary.
- It can be installed in Indoor (UMA) or outdoor (Castelliolí) environments as required by the project.

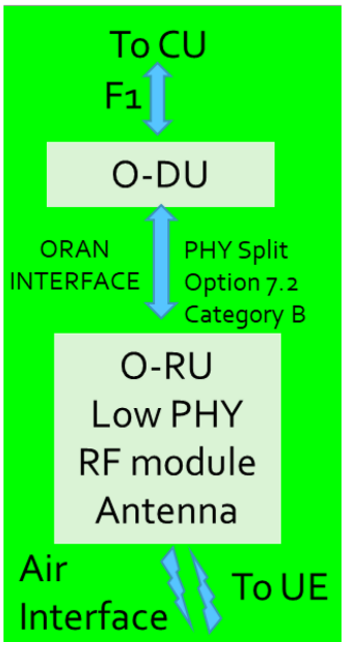


Figure 23: RU Interfaces

The Affordable5G O-RU is further detailed in Affordable5G D3.1 §2.1.2.

Table 5: O-RU interfaces

O-RU	**O-RAN conformant SMO APIs**	**Alternative scripting capability**
Lifecycle/Orchestration	Not Applicable	Not Applicable
Configuration (FCAPS)	Via the O-DU front-haul using Hierarchal Configuration. Ref. O-RAN O1 API [20] §2.1	Proprietary
Monitoring	Using NETCONF. O-DU works as NETCONF client and O-RUs as NETCONF server.	Proprietary

3.3.4 Near-RT RIC (dRAX)

The Affordable5G Near-RT RIC (RAN Intelligent Controller) is aligned to the O-RAN Alliance WP3 architecture [25] shown below in Figure 24.

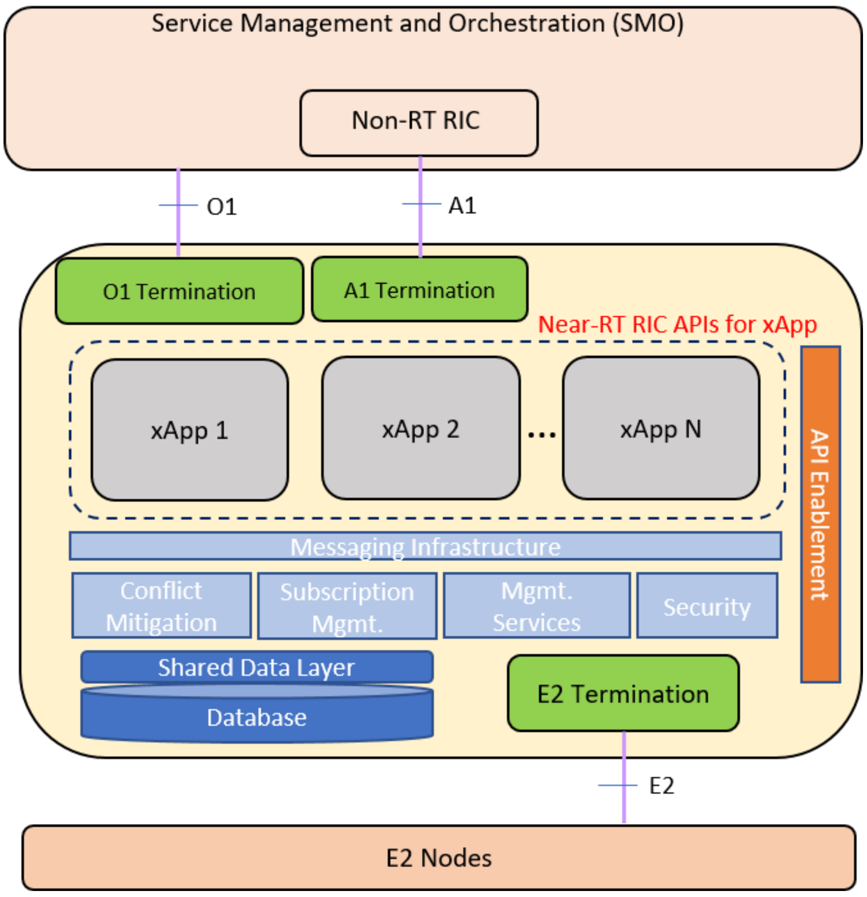


Figure 24: O-RAN WG3: Near-RT RIC Internal Architecture

The Near-RT RIC hosts the following functions:

- Database, which allows reading and writing of RAN/UE information
- xApp subscription management, which merges subscriptions from different xApps and provides unified data distribution to xApps
- Conflict mitigation, which resolves potentially overlapping or conflicting requests from multiple xApps
- Messaging infrastructure, which enables message interaction amongst Near-RT RIC internal functions
- Security, which provides the security scheme for xApps
- Management services:
  - Fault management, configuration management, and performance management as a service producer to SMO.
  - Life-cycle management of xApps.
  - Logging, tracing and metrics collection, which capture, monitor and collect the status of Near-RT RIC internals and can be transferred to external system for further evaluation
- Interface Termination:
  - E2 termination, which terminates E2 interface from an E2 Node.
  - A1 termination, which terminates A1 interface from Non-RT RIC.
  - O1 termination, which terminates O1 interface from SMO
- Functions hosted by xApps, which allow services to be executed at Near-RT RIC and outcomes to be sent to E2 Nodes via E2 interface.
- API enablement function supporting capabilities related to Near-RT RIC API operations (API repository/registry, authentication, discovery, generic event subscription, etc.)

Table 6: Near-RT RIC interfaces

Near-RT RIC	O-RAN conformant SMO APIs	Alternative scripting capability
Lifecycle/Orchestration	O-RAN O2 API [19]	Helm charts, [21]
Configuration (FCAPS)	O-RAN O1 API [20] §2.1	Helm charts, [21] (no dynamic reconfiguration, restart needed)
Monitoring	Via xApp telemetry exporter (Kafka/Prometheus), emulation of O1 ONAP VES	Rest APIs

3.4 AI/ML Framework

The AI/ML framework proposed by Affordable5G is an important building block of its architecture. The presence of this block provides Affordable5G with the capability to integrate AI and ML algorithms into models that can serve the inference requests from diverse control loops across other building blocks. Slice Infrastructure resource prediction and near-Real-Time RIC local execution of models as xApps, are two examples of the use cases where the AI/ML framework can operate within Affordable5G.

Figure 25 (taken from the Linux Foundation) shows the three control loops proposed by the O-RAN Alliance:

- **Real-time control loop (< 10 ms)**, where the DU can directly change the radio resource management policies and quickly react on potential issues affecting the user QoE.
- **Near-RT control loop (>= 10 ms and < 1 second)**, where the near-RT RIC enable eNB/gNB functionalities as xApps on northbound interfaces, including mobility management, admission control and interference management. These functions are available as Apps on the controller, which enforces network policies via south-bound interfaces towards the radios. Furthermore, these control functions rely on analytics and data-driven approaches, including advanced ML/AI tools to enhance resource management capabilities.
- **Non-RT control loop (>= 1 second)**, where the non-RT RIC enables intent-based management based on the principles of automation and AI/ML, setting policies per user and data enrichment information for RAN optimization. Similarly, to the near-RT RIC, with the non-RT RIC is possible to build an eco-system of intelligent controller software where applications (rAPPs) feed each other with data and insights. Furthermore, the non-RT RIC introduces the ability to send to the near-RT RIC optimization policies targeting an individual user through the new A1 interface, complementing the per-node configuration interface O1.

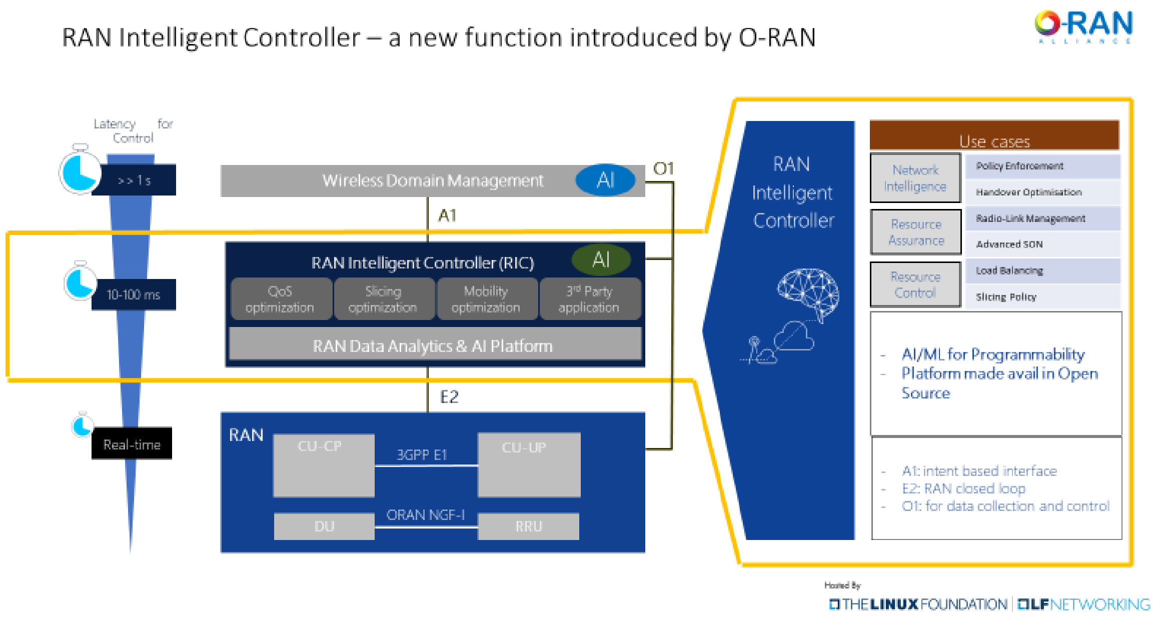


Figure 25: O-RAN Alliance control loops

Figure 26 shows the integration of an AI/ML model in the Near-RT control loop. The model is prepared using the AI/ML framework and passed on to the near-RT RIC for its orchestration as an xApp. The xApp Processor is in charge of all the logic related to the inference procedure. This module, reads data from the infrastructure published by the RIC on the databus, places an inference request to the model hosted on a model server running within the xApp, and writes the result on another topic in the databus for other xApps to consume.

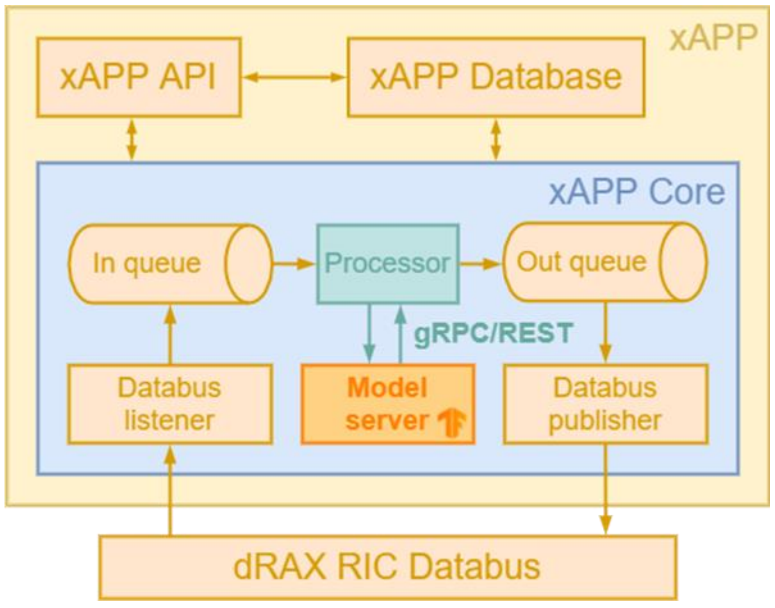


Figure 26: TensorFlow Serving in xAPP

Figure 27 depicts the components that take part in ML loop for slice infrastructure resource prediction. In particular, it can be seen that ML predictor is deployed on a separate container imported from the ML model server and is periodically fed with the information of the infrastructure resources of the slices that the orchestrator makes periodically available to the telemetry data collector. With this data, the ML predictor places an inference request to the model hosted on a model server and, based on the response received, acts towards the orchestrator via a message broker.

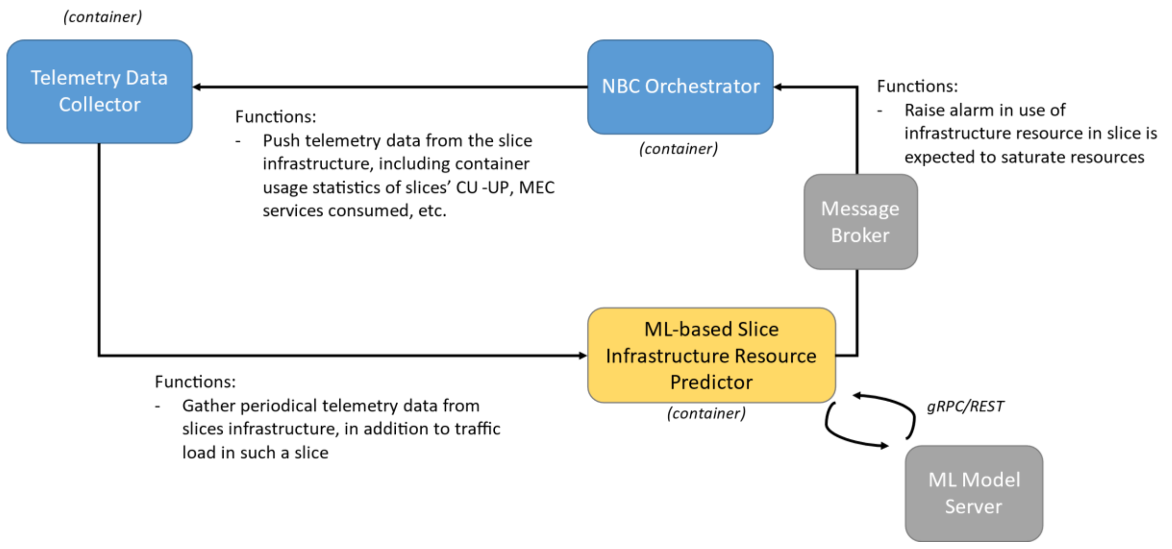

(container)
Telemetry Data Collector
NBC Orchestrator
(container)
Functions:
- Push telemetry data from the slice infrastructure, including container usage statistics of slices' CU -UP, MEC services consumed, etc.
Functions:
- Raise alarm in use of infrastructure resource in slice is expected to saturate resources
Message Broker
ML-based Slice Infrastructure Resource Predictor
(container)
Functions:
- Gather periodical telemetry data from slices infrastructure, in addition to traffic load in such a slice
gRPC/REST
ML Model Server

Figure 27: Integration plan of the ML loop for slice infrastructure resource prediction

The framework allows the AI/ML developer to design and orchestrate workflows based on TensorFlow eXtended (TFX) [26] libraries, forming complete pipelines, including data ingestion, statistics generation, validation, transformation of data, training, evaluation and finally serving of AI/ML models. These pipelines can be scheduled to be run automatically so that new versions of the models can be served after being trained with the latest data available at discretion of the developers.

There are three main functions performed by the AI/ML framework:

- **Pipeline Orchestration:** Orchestration of pipelines that cover all steps in the process from data ingestion until the output of a new version of the model.
- **Model Discovery:** Scanning for new versions of the models as outputs of the pipeline orchestrator to trigger the necessary steps for serving the model.
- **Serving Automation:** Following the discovery of a new version of a model, this module automates the placement of the new model on a TensorFlow Serving instance (a container-based model server included in TensorFlow libraries) that is available for inference from the relevant control loops.

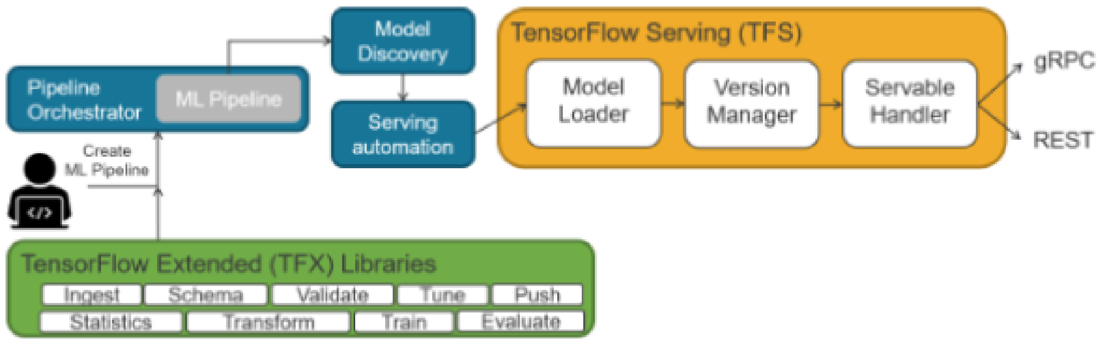


Figure 28: Affordable5G AI/ML Framework architecture

The main interfaces relevant to the AI/ML framework are as follows:

- The AI/ML pipelines will be orchestrated using **Apache Airflow**. Apache Airflow allows to run, schedule and monitor AI/ML workflows as Direct Acyclic Graphs (DAGs). The setup of these DAGs is realized by means of a **python module included in TensorFlow Extended libraries** [26]. Complementarily, the status of current DAGs can be queried at any time using **Apache Airflow Python Client** [27], **CLI** [28] or **REST API** [30].
- To cater for inference requests from the clients, **TensorFlow Serving** creates an endpoint for each model and version in the form of either **REST API** [30] or **gRPC** [31] requests.
- Lastly, in the execution of models as xApps use case, the near-Real-Time RIC exposes an **xApp lifecycle management API** that may be consumed by the AI/ML framework to place the models. The near-Real-Time RIC block is described in section 3.3.4.

	Apache Airflow	TensorFlow Serving	near-Real-Time RIC
Orchestration	Orchestration of AI/ML Pipelines as DAGs	N/A	Orchestration for AI/ML models as xApps
Monitoring	Monitor status of scheduled DAGs	N/A	Monitor Status of deployed xApps
Testing	Verify correct setup of DAGs	Verify availability of models for Inference	Verify correct deployment of models as xAPPs

Some aspects on the AI/ML framework block can be tested using OpenTAP tool:

- Verify correct setup of DAGs
- Verify availability of models for Inference
- Verify correct deployment of models as xAPPs

Command Name	Description	Type	External Interface Status
CREATE	Create AI/ML pipeline	Shell/Python Script	Available (Airflow Interfaces & TFX libraries)
SCHEDULE	Verify AI/ML pipeline scheduling	REST API, CLI, Python-Client	Available (Airflow Interfaces)
OUTPUT	Verify model output by DAG.	Shell Script	Available
INFERENCE	Verify model availability and response to inference request	REST API, gRPC	Under development
PLACEMENT	Perform placement of model as xApp on near-Real-Time RIC	REST API	Available (Near-RT RIC API)
REMOVE_xAPP	Remove xAPP from near-Real-Time RIC	REST API	Available (Near-RT RIC API)
REMOVE_DAG	Remove DAG scheduling from orchestrator	Shell/Python Script	Available (Airflow Interfaces & TFX libraries)

3.5 MANO for Services – NBC

NearbyComputing is providing Affordable5G with NearbyOne, a Multi domain e2e-orchestrator aligned with the project's requirement of a MANO for services.

Through our dashboard interface (there's also an OSS interface that in principle wouldn't be used) the user can define the requirements to deploy the VNFs/Apps in terms of Customer driven Requirements (e.g., multi-tenancy), Location driven requirements (security, power, distance or geographical area) and Service driven Requirements (SLA, reliability, …). The NearbyOne orchestrator hosts the following components/functions:

- Secure provisioner, provisions the base OS and hardware/software/accelerators configuration required on the managed edge nodes.
- Monitoring of KPIs from the infrastructure, network components, VIM, VNF/Apps.
- SLA Manager keeps track of the SLA defined by the user and the KPIs gathered and triggers the pre-defined actions when SLA thresholds are triggered.
- Service Placement Manager, the user can either decide where the VNF/Apps are deployed or can let the Service Placement Manager select the better location and migrate it accordingly to keep the SLAs.

The NearbyOne orchestrator will integrate with i2CAT Slice manager for the RAN slice chunks and manage internally the rest of the slice chunks.

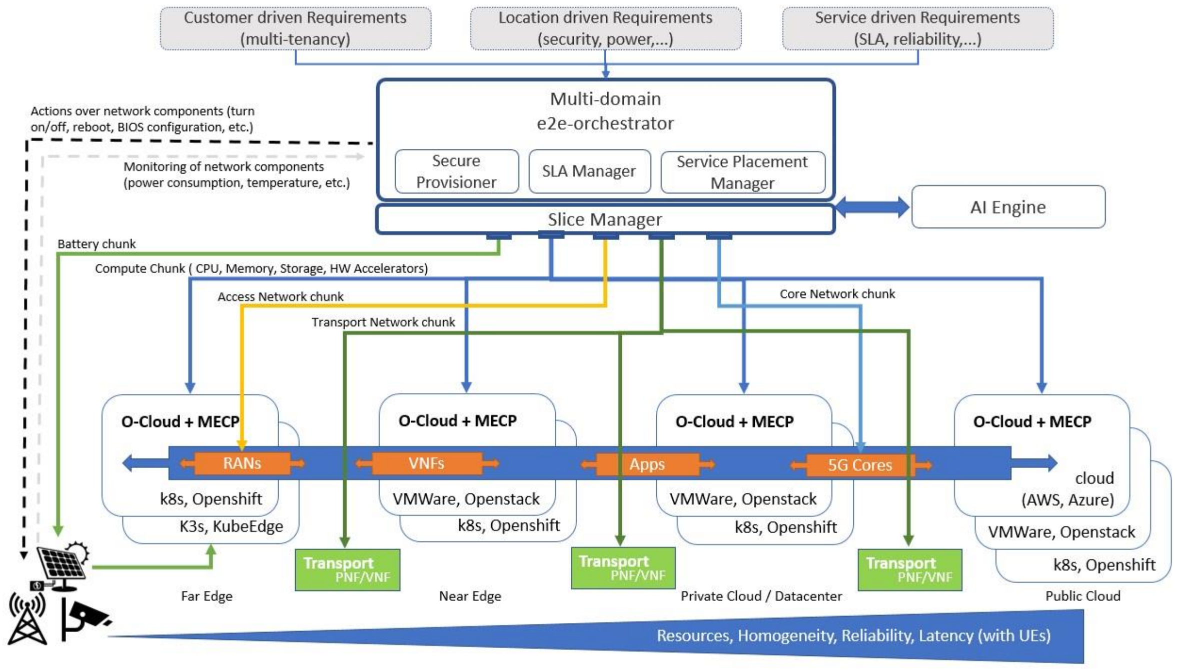

Customer driven Requirements (multi-tenancy)
Location driven Requirements (security, power,...)
Service driven Requirements (SLA, reliability,...)
Multi-domain e2e-orchestrator
Secure Provisioner
SLA Manager
Service Placement Manager
Slice Manager
AI Engine
Actions over network components (turn on/off, reboot, BIOS configuration, etc.)
Monitoring of network components (power consumption, temperature, etc.)
Battery chunk
Compute Chunk (CPU, Memory, Storage, HW Accelerators)
Access Network chunk
Transport Network chunk
Core Network chunk
O-Cloud + MECP
RANs
VNFs
Apps
5G Cores
k8s, Openshift
K3s, KubeEdge
VMWare, Openstack
cloud (AWS, Azure)
Transport PNF/VNF
Far Edge
Near Edge
Private Cloud / Datacenter
Public Cloud
Resources, Homogeneity, Reliability, Latency (with UEs)

Figure 29: NearbyOne architecture

3.6 MANO for Services – MARTEL

There are two enhancements to OSM (Open Source MANO) [32] that are planned. The first is to extend OSM's KNF (Kubernetes-based Network Functions) onboarding options to enable the placement of Kubernetes (K8s) pods on specific nodes of a registered K8s cluster. The second is to introduce an infrastructure as code approach to the onboarding of the KNFs, allowing for the descriptors to be stored in a Git repository and sync them to the actual deployment in the K8s clusters.

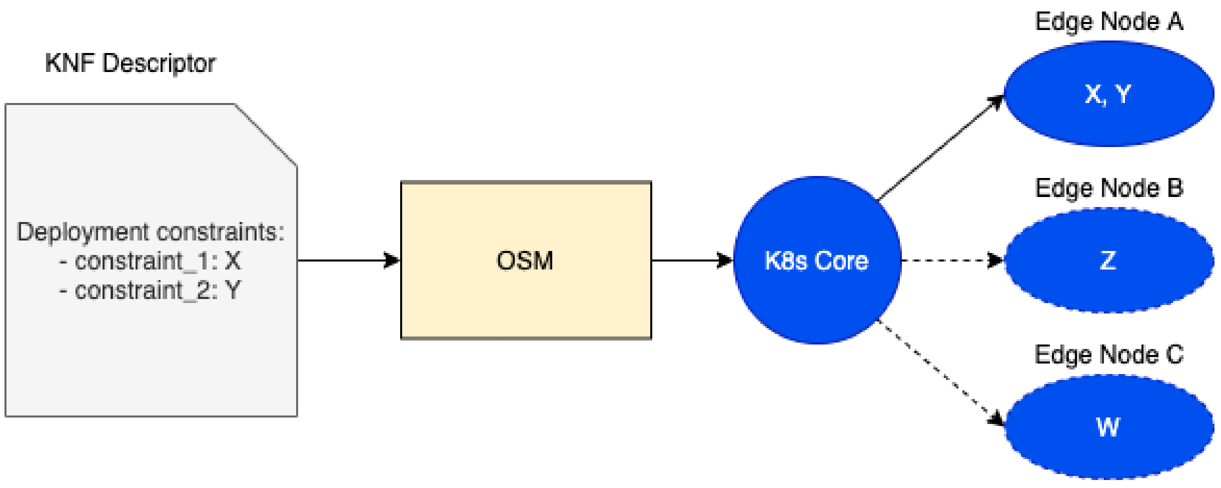


Figure 30: OSM's KNF enhancement

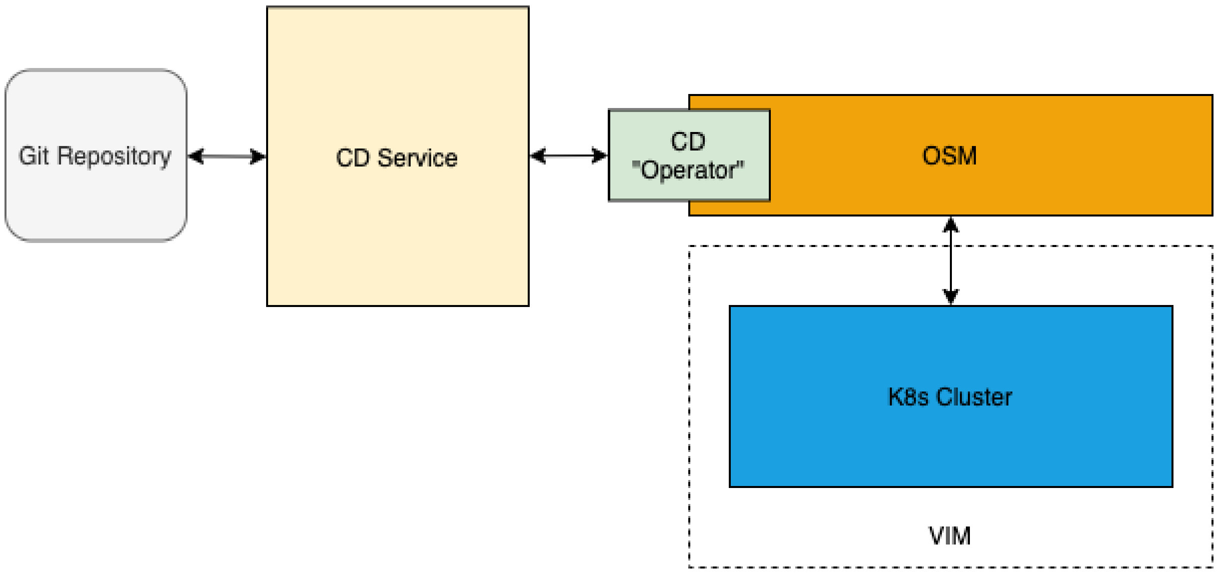


Figure 31: Introduction of an infrastructure as code approach to KNFs

Neither feature could support the use of OpenTAP for testing and integration. In the placement feature case, there is an OpenTAP plugin that is capable of creating and deleting VNFs on OSM, but there is no way to test against Kubernetes to check if the pods associated with the given KNF are deployed on the correct node. OSM itself does not account for that level of granularity with Kubernetes clusters, and OpenTAP doesn't allow for testing with OSM primitives, or directly from Kubernetes either. Allowing for OpenTAP to do this would require a new extension to the OpenTAP plugin, or developing a new one altogether, both of which is beyond the scope of the work. Given that the ultimate goal of the placement enhancement is to have it included with OSM as an officially supported feature, the eventual testing and integration would need to fit within OSM's already existing testing and integration setup (using Jenkins). In the context of Affordable5G a possible solution would be to use a "simpler" CI (Continuous Integration) solution to check against Kubernetes' API for the placement.

For the infrastructure as code feature, because the process of deploying and syncing from descriptors in a repository is handled by a separate service external to OSM, testing with OpenTAP also does not apply for similar reasons as the placement feature. This feature would also need to be tackled through a separate CI solution, if possible, alongside the placement feature.

3.7 Telemetry

Network Telemetry can be viewed as a real time data collection from various components of the architectural layers. In this context, two significant components are the NWDAF (Network Data Analytics Function) and the C-MDAF (Centralized Management Data Analytics Function), shown in Figure 1, where the first one is a well-specified component of the 5G Core network according to 3GPP specifications. The NWDAF collects data from and provides network data analytics services to 5GC NFs. The C-MDAF is provisioned with all the centralized telemetry capabilities, located at the Management, Orchestration and Automation layer. In this context, a particular NF (Network Function) can subscribe to the C-MDAF as a consumer in order to collect or provide management data for forecasting or resource information purposes.

The overall telemetry approach, described in section 3.3.5 of D1.2 [33] and in section 3.4 of D3.1 [7], can be seen in Figure 32. The 5GS Telemetry Data Collector, which is integrated with the C-MDAF, includes a monitoring server (Prometheus) for collecting performance measurement data from the NWDAF, the virtualized infrastructure and the Transport Network elements. Performance measurement data are also collected from the O-RAN VNFs using the O1 Virtual Event Streaming (VES) collector. The NWDAF incorporates the necessary interfaces to collect data from different types of data sources, notably 5G Core NFs. These data are made available to a Prometheus monitoring server residing in the NWDAF. Both the NWDAF and the C-MDAF provide data analytics; in this respect, ML-based data analytics are provided by the ML analytics module (that can be regarded as part of the AI/ML framework) shown in Figure 32.

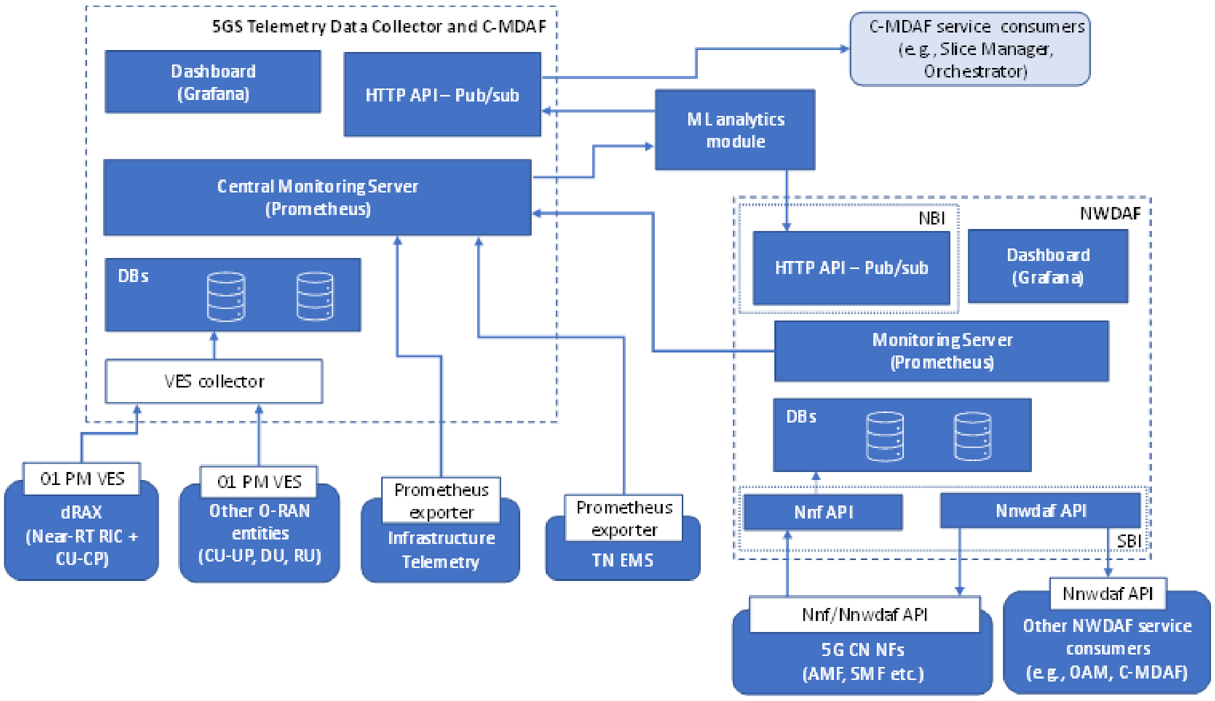

5GS Telemetry Data Collector and C-MDAF
Dashboard (Grafana)
HTTP API – Pub/sub
Central Monitoring Server (Prometheus)
DBs
VES collector
C-MDAF service consumers (e.g., Slice Manager, Orchestrator)
ML analytics module
NBI
NWDAF
HTTP API – Pub/sub
Dashboard (Grafana)
Monitoring Server (Prometheus)
DBs
Nnf API
Nnwdaf API
SBI
O1 PM VES
dRAX (Near-RT RIC + CU-CP)
O1 PM VES
Other O-RAN entities (CU-UP, DU, RU)
Prometheus exporter
Infrastructure Telemetry
Prometheus exporter
TN EMS
Nnf/Nnwdaf API
5G CN NFs (AMF, SMF etc.)
Nnwdaf API
Other NWDAF service consumers (e.g., OAM, C-MDAF)

Figure 32: Overall telemetry approach

The telemetry implementation consists of a set of containers which are self-configured, so there is no need for further configuration actions undertaken by the OpenTAP. However, appropriate scripts are provided and used to perform tests that can be executed from the OpenTAP in order to validate the deployment of the telemetry component. The deployment control functionality that can be exploited by the OpenTAP tool is presented in Table 7.

Table 7: Telemetry deployment control functionality

Functionality	**Provided through**	**Notes/Output**
NWDAF Deployment	Bash script	Script returns 0 on successful deployment of the containers
Check that the containers are up and running	Bash script	Script returns 0 if all NWDAF containers are running
Check the NWDAF API	Bash script	Script returns 0 if the NWDAF API is running
Check Monitoring services	Bash script	Script returns 0 if Monitoring services are running
Reconfiguration of the monitoring server	Bash script	Actions: - Reconfigure Monitoring server - Check default configuration - Change server configuration and restart server - Check new configuration Script returns 0 on successful execution

Preliminary integration and testing of the telemetry and AI/ML components for O-RAN optimization, using a Deep Reinforcement Learning (DRL) based approach exploiting xAPPs, is presented in the Appendix.

3.8 Time sensitive networking over 5G + PoC TSN – ADVA-REL

To evaluate TSN over 5G network, ADVA is planning to build the test setup depicted below.

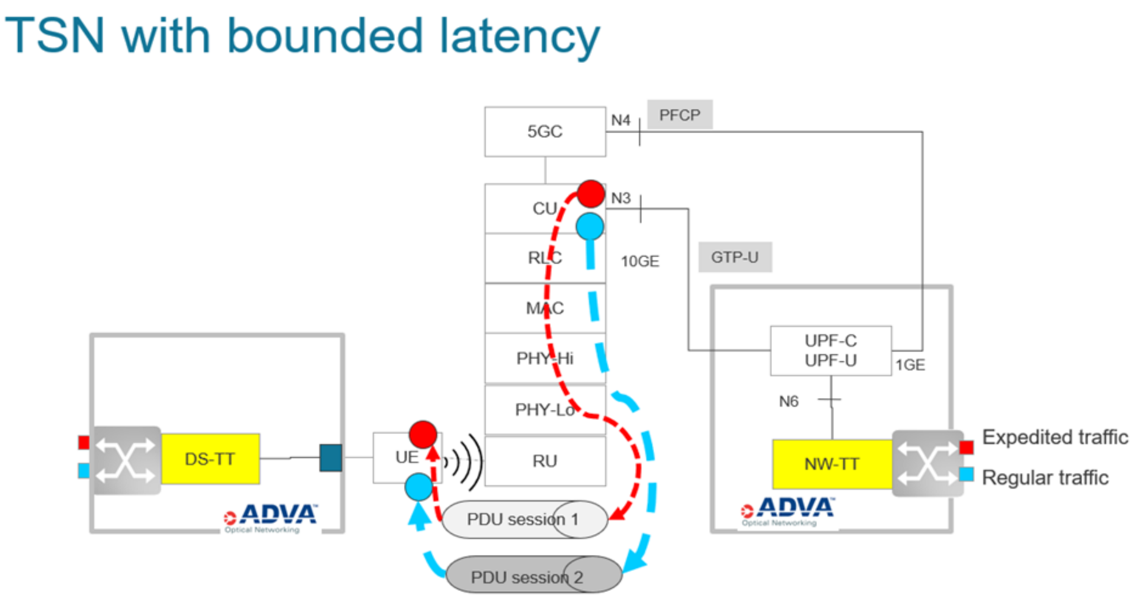


Figure 33: TSN over 5G test setup (ADVA)

We are investigating the 5G NPN behaviour with regard to two traffic types: Expedited and Regular.

Expedited traffic path is marked in red. It represents jitter-sensitive control traffic, where packets are short and expected to be delivered within limited time window. The traffic is injected to a dedicated 1GE port to the Network TSN-Translator (NW-TT) function. The NW-TT performs some packets rearranging and sends it towards the UPF. The Ethernet link between the TT and the UPF is either 10GE or 2x 1GE, depending on equipment availability. The UPF performs mapping two traffic streams into corresponding QoS flow, encapsulates it into the GTP-U and sends towards the CU-UP over the N3 reference point. Several options for the QoS mapping will be evaluated: two pre-established PDU sessions, single PDU session with different QoS flows etc.

The link between UPF and CU will be presumable 10GE. The CU-UP maps QoS flows into DRB. The traffic passes though 5G NR protocol stack (SDAP/PDCP/RLC/MAC) and arrives to the UE. The UE is connected to the DS-TT via a 1GE/10GE link. The traffic is processed by the DS-TT and transmitted through two 1GE ports to the data network.

The NW-TT has embedded traffic generator and analyser. Having fibre connected between the egress of the DS-TT and ingress of the NW-TT, we can measure one-way latency, jitter and packet loss.

Table 8: ADVA-REL TSN interfaces

	DS-TT	**NW-TT**	**UPF**
Control	N/A	N/A	Enable/Disable service
Configuration	cfg file	cfg file	cfg file
Monitoring	Throughput/ Packet loss	Throughput/ Packet loss	Throughput/Delay/ Packet loss

3.9 Time sensitive networking over 5G + PoC TSN – UMA-ATH

The current TSN infrastructure is composed by different elements that act as key enablers to get a first TSN setup at UMA. Mainly these elements are:

- TSN translators, DS-TT and NW-TT. UMA will provide TSN translators needed to enable the communication between TSN and 5G domains.
- TSN bridge. This element acts as a switch supporting the TSN features.
- TSN end station. These elements are the end of the communication, sometimes called talker and listener, depending on the direction of the TSN flow are the source and destination of the TSN flow.
- CNC. Application in charge of the network configuration determining the routes and scheduling of TSN flows.
- CUC. Application connected with the CNC that receives the requests from TSN end stations for deterministic communication.

The deployment at UMA includes the Relyum solution [34] that includes 1 TSN bridge and 2 TSN PCIe cards based on Intel I210 providing IEEE 802.1 TSN contrasted features able to act as a bridge or end station. In addition, 4 Intel I210 cards are used as end stations for transmitting and receiving TSN traffic. Related to TSN translators, the U-Blox EVK-R5 solution [35] allows to get the network clock needed for synchronization mechanisms with TSN clock.

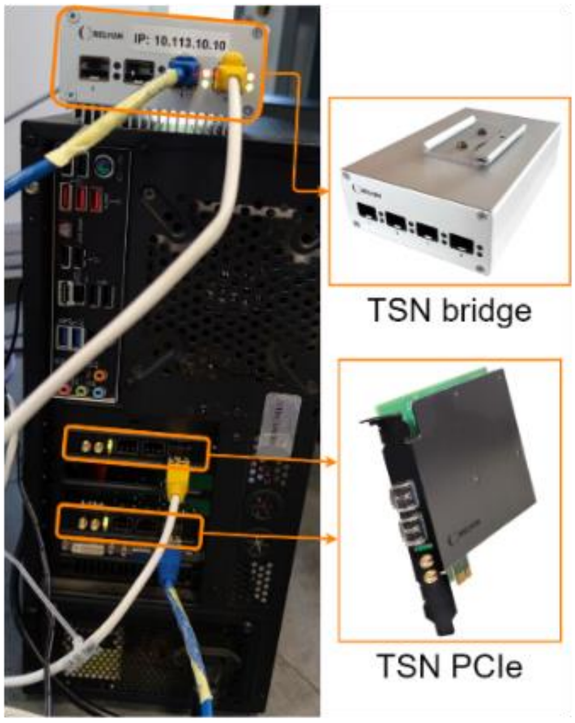


Figure 34: Relyum setup

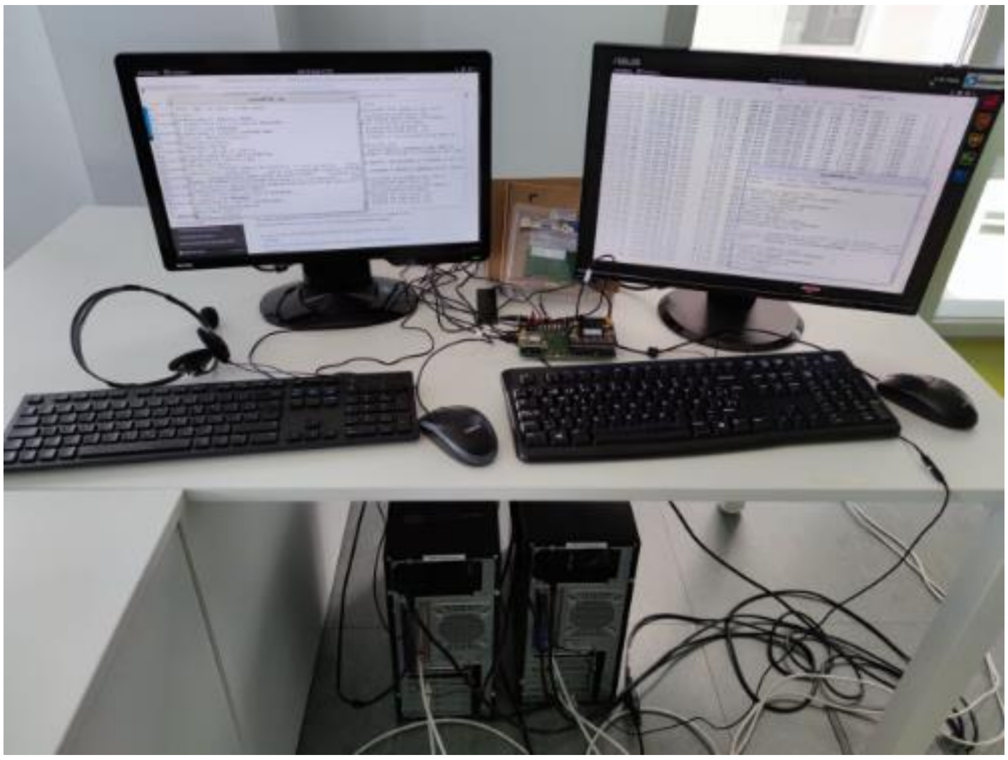


Figure 35: U-blox and Intel I210 cards

Table 9: UMA-ATH TSN interfaces

	DS-TT	NW-TT	CNC-CUC	AF-TSN
Control	Synchronization	Synchronization	Synchronization	Enable/Disable service
Configuration	Scheduling table	Scheduling table	Apply configuration / Change parameters value (TSN switches & end-station)	IP address/ IP route/Resources resize
Monitoring	Throughput/ Delay/ Packet loss	Throughput/ Delay/ Packet loss	Current configuration/ Requested KPIs	CPU/Memory/ Disk/Throughput

3.10 Smartcity service

The Computer Vision Analysis for Emergencies (CVAE) service relies on multiple components and logical functions installed both in the edge and the core of the network. The CCTV cameras deployed at the edge play an essential role as they represent the main input of data from which the CVAE service will operate. As described in previous deliverables, one of the main objectives of the CVAE service is to provide a real-time analysis of the images to the purpose of detecting emergency events. To this end, some key network requirements must be assessed. Using automated testing it is possible to determine the value of key KPIs to ensure the proper execution of the CVAE service, namely the bandwidth, the latency and packet loss. Within the CVAE architecture for the core network, there is an exposed API for configuration of parameters both on edge devices (such as CCTV cameras) and edge CVAE submodules, represented bellow in Figure 36.

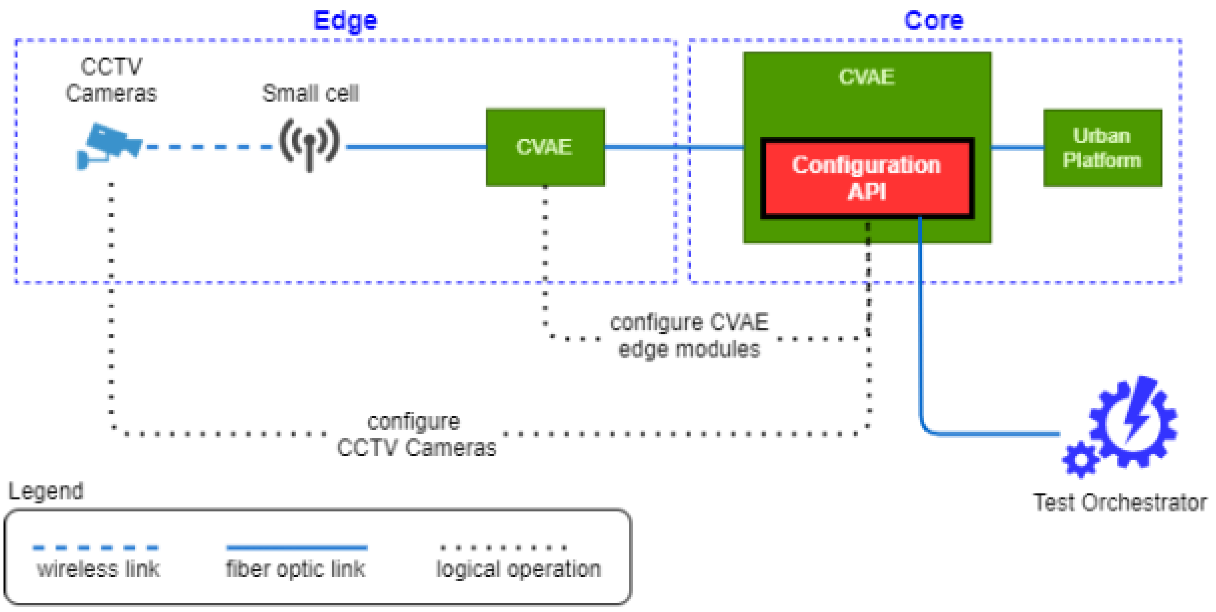


Figure 36: CVAE Configuration Module

An external entity, here represented as the Test Orchestrator, can configure the components of CVAE service through the Configuration API to suit the programmed experimental tests. Furthermore, the Configuration API includes routines for the purpose of testing (development mode) that allow the triggering of emergency scenarios even when there are none occurring in the real scenario (dummy triggering). The functionalities of the Configuration API are represented in Table 10.

Table 10: CVAE test functionalities

	CCTV Cameras	**CVAE (edge)**	**CVAE (core)**
Control	N/A	N/A	Enable/Disable dev testing routines
Configuration	Resolution, FPS, Codec	Frequency of detection analysis, Frequency of classification	Apply configurations / Change parameters on edge devices and modules
Monitoring	Throughput/Latency/ Packet loss	Throughput/Latency/ Packet loss	Retrieval of requested KPIs

In summary, the Test Orchestrator will have the ability to enable the development testing routines (pre-programmed in CVAE service), configure parameters of devices and modules of the service at the edge (or edges, if multiple are available), trigger the execution of the tests and, finally, retrieve the requested KPIs. The most important KPIs to assess if the service is working properly are the throughput, latency and packet loss between the CCTV camera and the edge computing functions and the throughput, latency and packet loss in the backhaul, linking the edge functions to the core functions of the CVAE service.

3.11 MCS service

The automated testing of the MCS will be done taking into account the end-to-end chain, thus, involving both the client side installed in Android UEs and the server side as the virtualized application function. To that end, we will emulate the behaviour of different MCS clients through the automation of testing steps and the server side will play the role of service provider and supporter without exposing any testing API to the outside world or automation tool. To better understand how the testing environment will be provided, we need to first discuss the architecture of the Nemergent MCS Client.

The MCS client itself implements internal interfaces and APIs so as to be compatible with other vendors building blocks. The complete MCS client architecture is shown in the Figure 37 below:

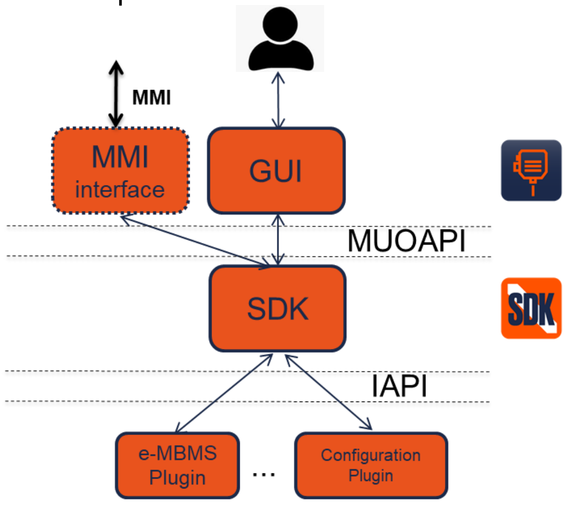


Figure 37: MCS Client architecture

- The SDK supports both APIs defined by the project MCOP(Mission Critical Open Platform) [29] - (both northbound MUOAPI - MCOP Unified Open Application API - as well as southbound IAPI - MCOP Integration API).
- The GUI supports the MUOAPI MCOP API. Nemergent GUI can include the extension of an MMI interface (Man to Machine Interface) that both supports communication with the SDK through the MUOAPI API and it is able to communicate with external executors through MMI commands. Therefore, through the MUOAPI API the SDK can both receive commands from the GUI with human interaction or through the MMI interface for automation tests.
- Finally, all plugins like the provisioning tool or the eMBMS middleware support IAPI MCOP API.

Once the MCS Client architecture and automation interface is clear, it is important to detail the testing topology (Figure 38) to understand that the automation of the MCS Client can test the whole MCS system end-to-end.

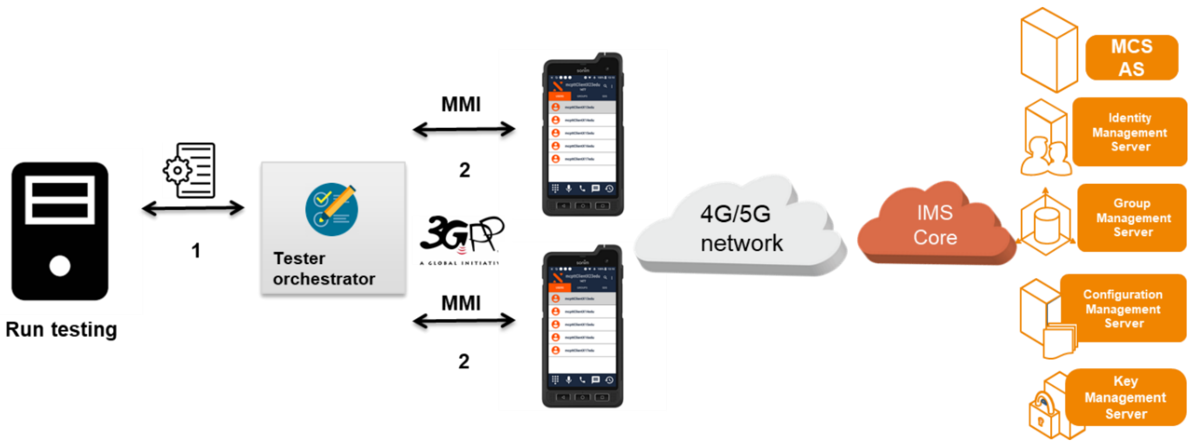


Figure 38: MCS system testing topology

When running specific tests with the shown architecture and topology, there will be a test script with all the sequences for specific testing (e.g., group calls). The test script is loaded in the tester orchestrator following the formatting rules of each tester orchestrator so, later on, it is easy for the tester to follow the plan and send commands through MMI to the UEs containing the MCS Client. Those MCS Clients will target the MCS system as a whole for the execution of each action testing this way the full system chain.

The service elements involved in the control, configuration, monitoring and testing of the service are the following ones (Table 11):

Table 11: MCS interfaces

	MCS AS	**MCS exporter**	**All MCS system**	**MCS Client**
Control	Service start/stop	N/A	N/A	N/A
Configuration	N/A	N/A	*e.g.,* new IP > whole systems' dockers will restart with the new configuration	N/A
Monitoring	N/A	Prometheus compliant interface to extract the following metrics: • MCPTT & MCVIdeo private calls	N/A	N/A

	MCS AS	MCS exporter	All MCS system	MCS Client
		• MCPTT & MCVideo private emergency calls • Nº of provisioned users and groups • MCPTT & MCVIdeo group calls • MCPTT & MCVIdeo group emergency calls • MCPTT group calls participants		
Testing	N/A	N/A	N/A	See Table 12 below

The testing commands are listed in the Table 12 below:

Table 12: MCS testing commands

Command Name	**Description**	**Type**	**External API status**
AUTH	Initiate MCPTT authorisation/configuration	MMI	Not exposed yet
IDMS	Provide user credentials: username and password	MMI	Not exposed yet
AFF	Affiliate to an MCPTT group	MMI	Not exposed yet
CALL_PRE	Request the establishment of an MCPTT on-demand pre-arranged group call	MMI	Not exposed yet
CALL_PRI	Request the establishment of an MCPTT private call	MMI	Not exposed yet
AFF_SUBS	Request to discover which groups a target user is affiliated to and to subscribe to affiliation status changes for that target user	MMI	Not exposed yet
EM_UP	Upgrade of the ongoing On-Demand Pre-arranged Group Call to MCPTT Emergency Group Call	MMI	Not exposed yet
EM_DOWN	Cancel the Emergency State	MMI	Not exposed yet
FC_REQUEST	Request the floor control	AT	Not exposed yet
FC_RELEASE	Release the floor control	AT	Not exposed yet
HANG	Hang up call	AT	Not exposed yet

3.12 Slice Manager

The i2CAT Slice Manager is a slicing engine that enables the programmatic partition of access network resources and IP network resources into several slices based on the requirements specified by a Mobile Network Operators (MNOs). By means of the slice manager MNOs can manage network slices in a dynamic manner, i.e., dynamic service provisioning over the resources available for the specific slice. The slice management can be achieved by means of three main technology drivers. 1) OpenStack, dealing with the Virtual Infrastructure Management; 2) Open-Source Mano (OSM), dealing with orchestration of NFVs (Network

Functions Virtualization) over Virtual Infrastructures; and 3) the non-real-time RIC, dealing with the management of the O-RAN.

The overall architecture of the slice manager can be observed in Figure 39. It is important to remark the Slice Manager handles requests (Step 3) by means of a RESTful API. Once a request is processed, the Slice Manager allocates the radio resources to the slice requested, as well as communicates with the non-RT RIC to allocate the radio resources in 4G (slices based on PLMNID) and 5G networks (slices based on S-NSSAI), c.f., step 4.

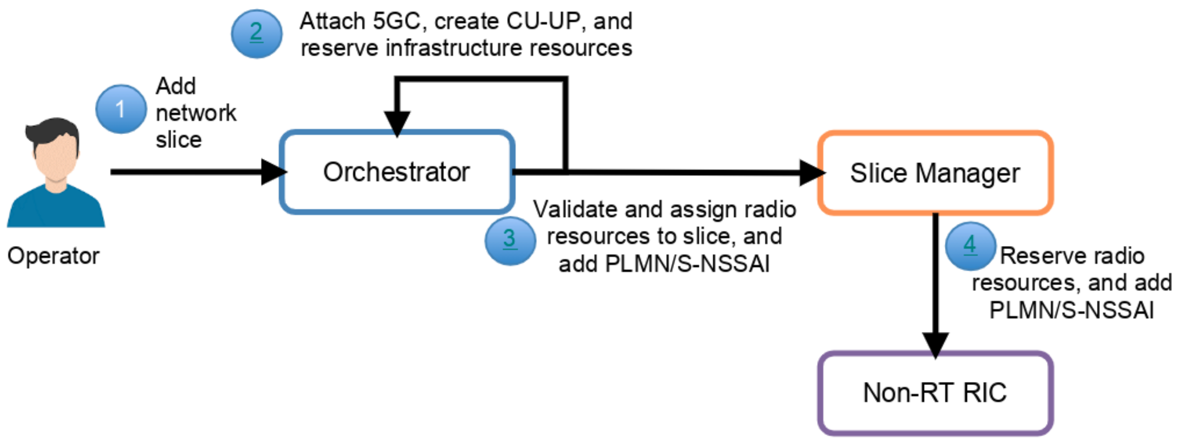

2
Attach 5GC, create CU-UP, and reserve infrastructure resources
1
Add network slice
Operator
Orchestrator
Slice Manager
3
Validate and assign radio resources to slice, and add PLMN/S-NSSAI
4
Reserve radio resources, and add PLMN/S-NSSAI
Non-RT RIC

Figure 39. Overall Architecture of the Slice Manager in Affordable 5G.

3.13 Non-RT RIC

Another RAN related management entity present in the Affordable5G architecture is the Non-RT RIC provided by i2CAT. This component implements two main functionalities:

- ORAN Element Management System (EMS). Provides REST endpoints that are used by the Non-RT RIC business logic to configure and provision services on top of the CU, DU and RU elements. Management of the previous functions is made available through the definition of YANG models that allow to interact with the devices using NETCONF. The ORAN EMS component of the Non-RT RIC implements a NETCONF manager function that translates NETCONF primitives to REST endpoints.

- The A1 policy management and A1 controller services. These components enable the management of xApp related policies in the Afforable5G Near-RT RIC component described in Section 3.3.4. In particular, the A1 policy management service discovers policies available in the Near-RT RIC and allows to create instances of those policies. The A1 controller service is the one in charge of communicating the policy instances to the Near-RT RIC using either a REST or NETCONF. The ability to deliver RAN related policies from the Non-RT RIC to the Near-RT RIC is a critical capability to enable the introduction of Machine Learning models that based on RAN telemetry can optimise the performance of the network.

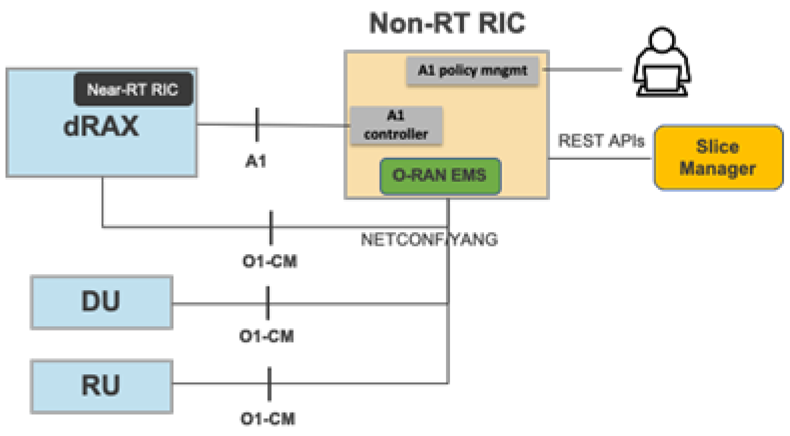


Figure 40.Affordable 5G Non RT-RIC with the EMS and AI controller component

4 INTEGRATION AND TESTING PLAN FOR CASTELLOLI

4.1 Castellolí platform and expected updates beyond the Affordable5G features

Castellolí platform has 9 Green Nodes already deployed, which are fully supplied with sustainable energy coming from solar panels and eolic turbines. These green nodes are already producing a full 4G coverage for the circuit.

The deployments that are expected to be realized in Castellolí are mainly focused in having 5G coverage across the circuit but keeping the energy consumption in the spotlight since the green nodes should keep working sustainably.

This will be possible by installing determinate HW that will give all the 5G functionalities, but also, to hold an AI mechanism that examines the sensor network and produce as an output the overall status of energy consumption per device connected into the green node. This will not just allow to ensure the correct operation of the node but will also enable the migration of critical services from nodes that are running out of energy before they actually turn off, and deploy this services into different nodes that will be able to hold this computing capacity. These actions will be led by the orchestrator, an element that will be deployed in the MEC.

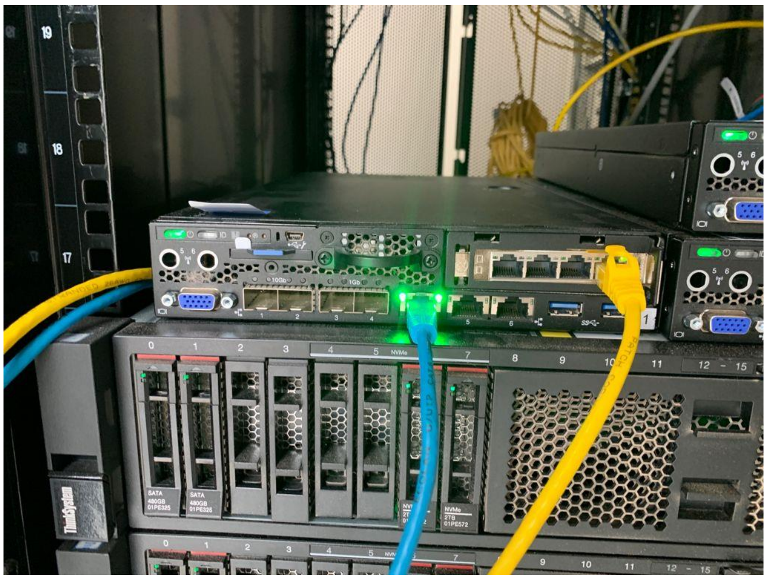


Figure 41: SE350 and SE650 located in the Rack

In the control room, a rack will be deployed containing:
- 2 Servers SE650, which will hold the 5GCore, the NearbyOne Orchestrator and the slice manager.
- 1 Palo Alto Firewall, which ensures the first protection layer and separates the different networks in Castellolí.
- 1 Server SE350, which holds the AI solution that allows to visualize the different consumptions per device and that will communicate with the orchestrator. This service is deployed in Kubernetes.

In the different 9 Green Nodes, the hardware that will be deployed consists of:
- 1 SE350 per Node, which will allow to deploy the different services in Kubernetes, and those services which require further functionalities will have accelerators cards that will ensure all the computing capacities will be covered.

4.2 Mapping details

Castellolí's architecture is depicted in Figure 42. The different building blocks are separated into blocks that will be deployed and the ones that have already been deployed.

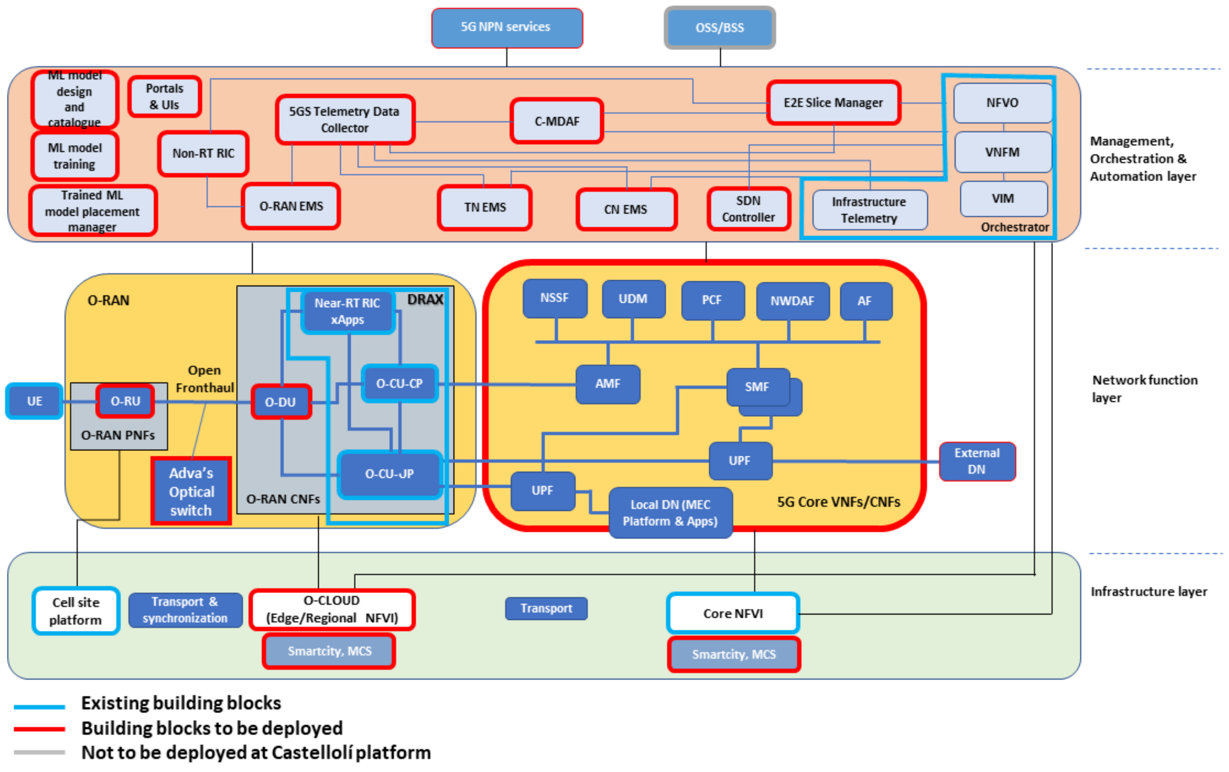


Figure 42: High Level architecture in Castellolí

The dRAX is deployed in the Lenovo SE650 server located in the Control Room in Castellolí. Additionally, all the Management and orchestration system will be also located in the same Lenovo machine

Figure 44: Lenovo ThinkSystem SR650 V2

Figure 43: Lenovo ThinkSystem SE350

All the servers and the different connectivity modules are placed in a rack that allocates all the security measures, such as the Palo Alto Firewalls.

The 5G SA Core provided by Athonet will also be deployed and installed in the Control Room.

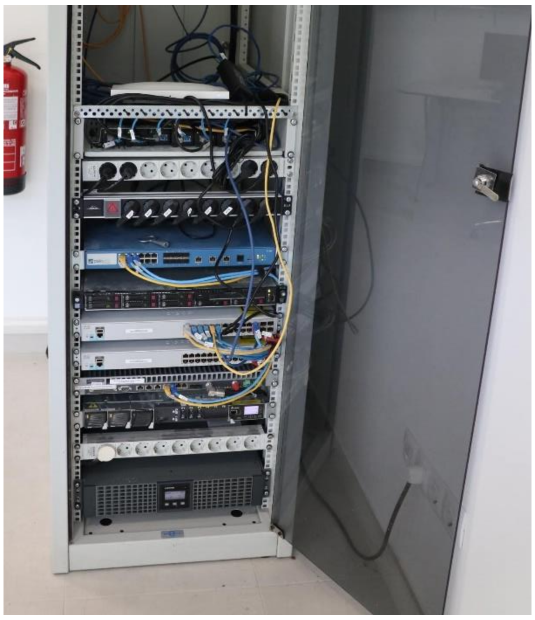


Figure 45: Control Room's Rack

The Control Room is connected to all the Edge Nodes in Castellolí, which sends the signals of the Energy status and the Video images to be displayed in the screens, so the activity in the circuit is analysed.

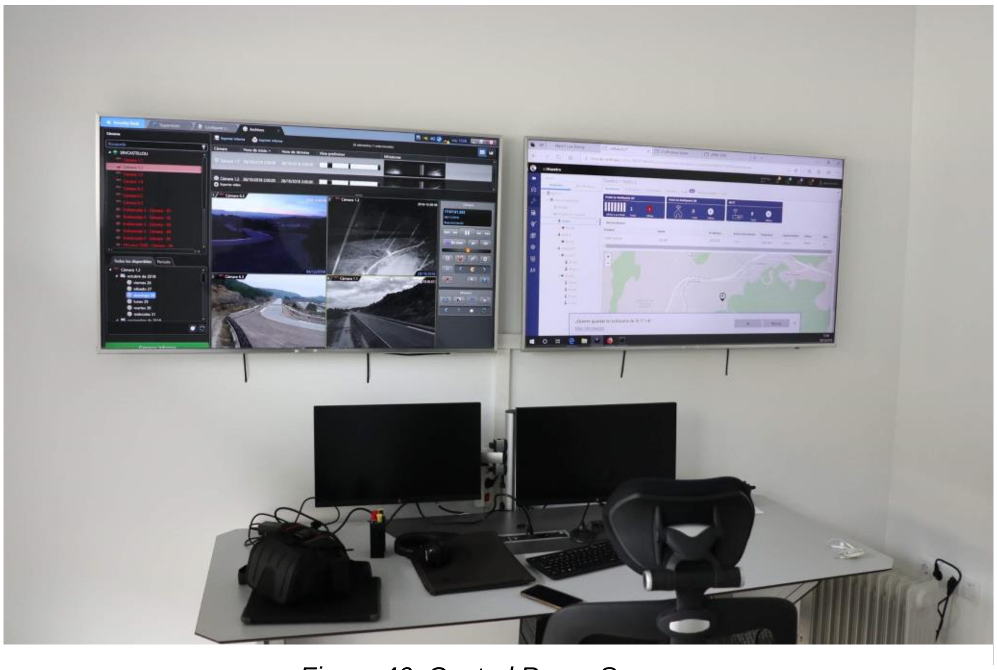


Figure 46: Control Room Screens

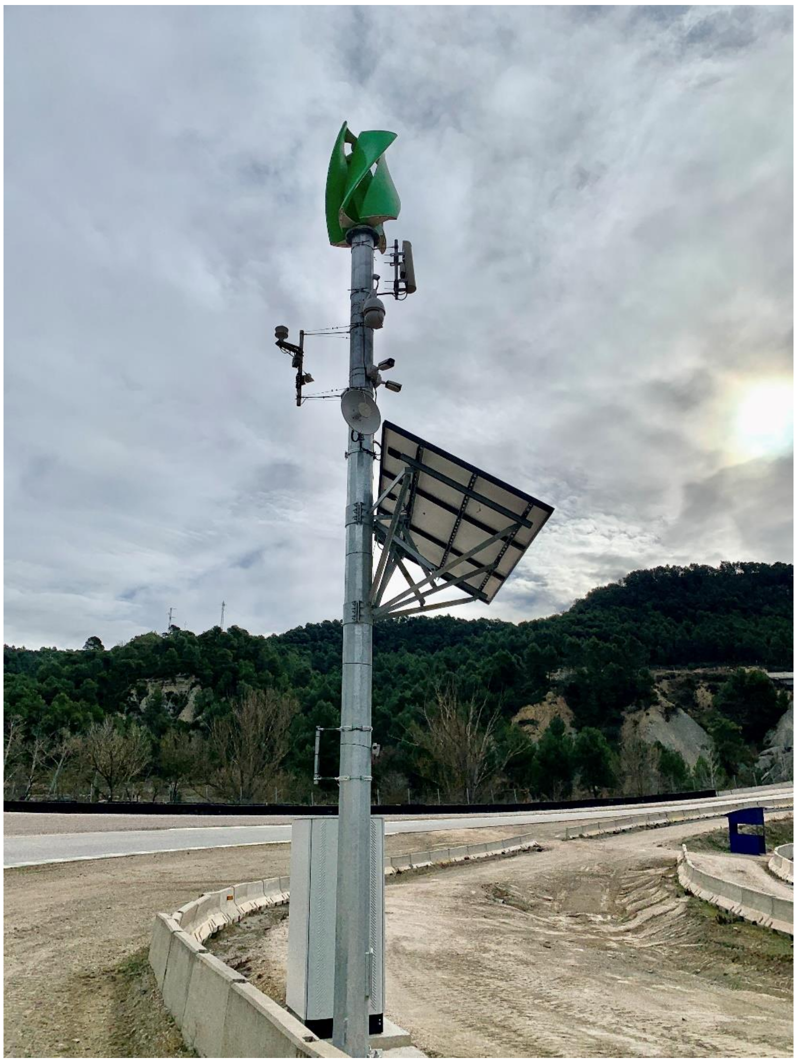


Figure 47: Edge Node

Figure 47 shows a self-sustainable edge node, comprising a cabinet and the HW components required for generating power. The energy is produced by the eolic turbine and the solar panels, and stored in the batteries located inside the cabinet. From the Control room, it is possible to check the battery status and the values of irradiance and wind speed in real time. These values are provided by a set of sensors also located in the cabinets.

The cabinets will also include the different SE350, that will have deployed the different services through Kubernetes.

4.2.1 Existing building blocks

In this subsection we describe the building blocks already deployed in the Castelloli test site.

Management and orchestration layer:
As part of the management, orchestration and automation layer, the NearbyOne orchestrator is already operational in Castelloli. Specifically, CELL and NBC have already started an integration, testing and validation activity with the goal of assessing the benefits of the orchestrator. For instance, the orchestrator is already capable of deploying the ACC dRAX and of switching on a 4G small cell.

Furthermore, a number of virtual machines has been made available for hosting other management components, including the slice manager, the telemetry data collector and the O-RAN EMS, which are already under validation.

Cell site platform:
As mentioned earlier, the Castelloli test site already features nine cell sites. One of them is connected to the power grid, whereas the others are self-sustainable. Furthermore, all the sites include a rack which can host additional computing machines.

dRAX:
The dRAX platform, i.e., collection of containers managed by Kubernetes that are responsible for handling the O-RAN CU-UP, CU-CP and near-RT RIC, is already available in Castelloli. However, the current version of dRAX only supports the deployment and management of 4G small cells.

4.2.2 New building blocks

In this subsection we describe the building blocks that will be deployed in the Castelloli test site.

O-RAN:
The whole 5G O-RAN block will be installed in Castelloli and will replace the existing 4G small cell provisioned by ACC. An upgraded version of the dRAX, i.e., 2.0, will be installed and managed by the NearbyOne orchestrator.
Moreover, to connect the RU and DU, the optical Fronthaul switch developed by ADVA will be deployed. Furthermore, the Precision Time Protocol (PTP) Grand Master (GM) clock will be installed for providing standard time information to other clocks across the network.
Finally, the EURE DU and the REL RU will complete the O-RAN deployment.

5GC:
ATH will provide a 5GC in a box which will be placed in the control room. Moreover, SIM cards will be provided to allow the authentication and access of 5G terminals.

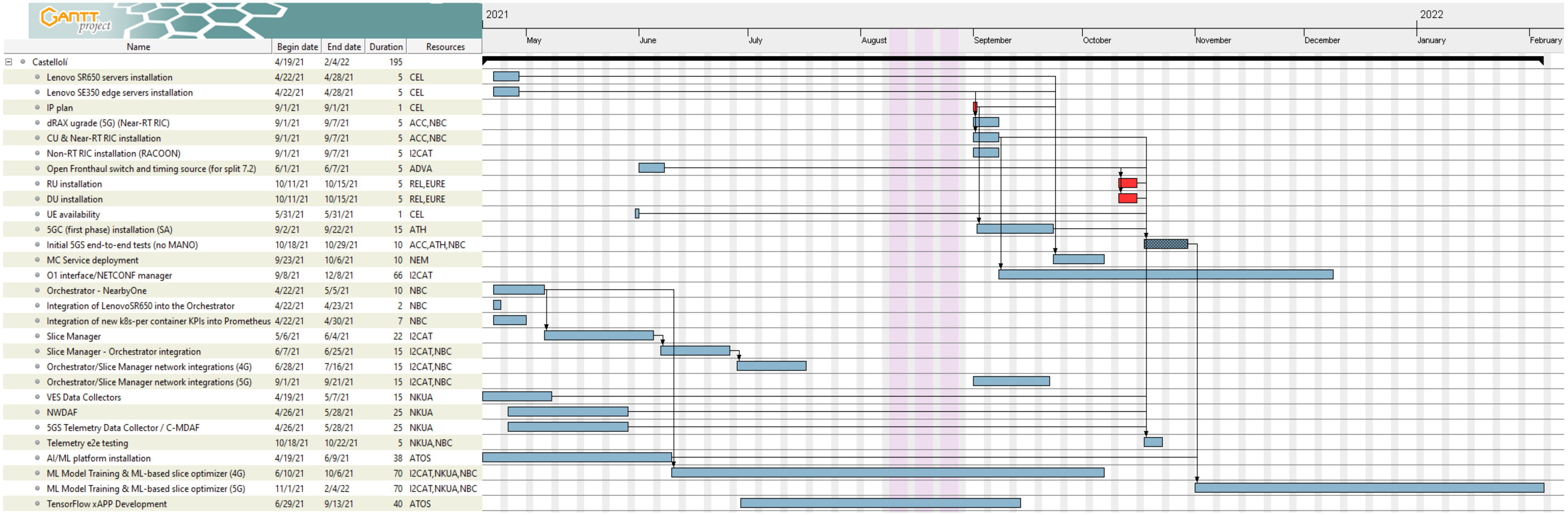
Management and orchestration layer:
Aside from the NearbyOne orchestrator, all the other key Affordable5G blocks of this layer will need to be deployed. Specifically, the non-RT RIC, the AI/ML platform and additional components handling the telemetry coming from the O-RAN and the 5GC will have to be installed and integrated in Castelloli.

O-Cloud:
The O-Cloud refers to the computing platform consisting of a collection of physical infrastructure nodes that can host the relevant O-RAN functions. Such platform has been already deployed in Castelloli and relies on two different computing machines, i.e., Lenovo ThinkSystem SE350 and Lenovo ThinkSystem SR650 V2. The first machine is a typical 1U rack edge server bringing computer power, storage and networking closer to the place where data is generated. In Affordable5G, the SE350 will be hosting the O-RAN DU block, which also connects with the RU and CU. The second machine is a 2U rack server based on the new third generation Intel Xeon Scalable processor family and hosts the dRAX and the orchestrator and management capabilities, including the NearbyOne orchestrator.
New computing machines will be potentially deployed based on the computing, networking and storage demand.

4.3 Planning for integration

The integration of all Infrastructure layer and Management layer elements in the two project sites involves many subtasks and dependencies. To better track the integration activity and identify possible bottlenecks, the consortium has developed and is maintaining a timeplan in the form of a Gantt chart.

The timeplan keeps track of all tasks, start dates, durations, dependencies between tasks and involved partners. The network integration activity is tracked separately for the two project test sites (Castellolí and Málaga). Below you can find the timeplan for Castellolí:



Name	Begin date	End date	Duration	Resources
Castellolí	4/19/21	2/4/22	195	
Lenovo SR650 servers installation	4/22/21	4/28/21	5	CEL
Lenovo SE350 edge servers installation	4/22/21	4/28/21	5	CEL
IP plan	9/1/21	9/1/21	1	CEL
dRAX ugrade (5G) (Near-RT RIC)	9/1/21	9/7/21	5	ACC,NBC
CU & Near-RT RIC installation	9/1/21	9/7/21	5	ACC,NBC
Non-RT RIC installation (RACOON)	9/1/21	9/7/21	5	I2CAT
Open Fronthaul switch and timing source (for split 7.2)	6/1/21	6/7/21	5	ADVA
RU installation	10/11/21	10/15/21	5	REL,EURE
DU installation	10/11/21	10/15/21	5	REL,EURE
UE availability	5/31/21	5/31/21	1	CEL
5GC (first phase) installation (SA)	9/2/21	9/22/21	15	ATH
Initial 5GS end-to-end tests (no MANO)	10/18/21	10/29/21	10	ACC,ATH,NBC
MC Service deployment	9/23/21	10/6/21	10	NEM
O1 interface/NETCONF manager	9/8/21	12/8/21	66	I2CAT
Orchestrator - NearbyOne	4/22/21	5/5/21	10	NBC
Integration of LenovoSR650 into the Orchestrator	4/22/21	4/23/21	2	NBC
Integration of new k8s-per container KPIs into Prometheus	4/22/21	4/30/21	7	NBC
Slice Manager	5/6/21	6/4/21	22	I2CAT
Slice Manager - Orchestrator integration	6/7/21	6/25/21	15	I2CAT,NBC
Orchestrator/Slice Manager network integrations (4G)	6/28/21	7/16/21	15	I2CAT,NBC
Orchestrator/Slice Manager network integrations (5G)	9/1/21	9/21/21	15	I2CAT,NBC
VES Data Collectors	4/19/21	5/7/21	15	NKUA
NWDAF	4/26/21	5/28/21	25	NKUA
5GS Telemetry Data Collector / C-MDAF	4/26/21	5/28/21	25	NKUA
Telemetry e2e testing	10/18/21	10/22/21	5	NKUA,NBC
AI/ML platform installation	4/19/21	6/9/21	38	ATOS
ML Model Training & ML-based slice optimizer (4G)	6/10/21	10/6/21	70	I2CAT,NKUA,NBC
ML Model Training & ML-based slice optimizer (5G)	11/1/21	2/4/22	70	I2CAT,NKUA,NBC
TensorFlow xAPP Development	6/29/21	9/13/21	40	ATOS

Figure 48: Timeplan for Castellolí

5 INTEGRATION AND TESTING PLAN FOR MALAGA

5.1 Malaga platform and expected updates beyond the Affordable5G features

The Malaga platform from UMA is constantly being expanded and improved with new and diverse equipment. Apart from the expected integrations and features that Affordable5G will bring into the platform, some other extensions are happening in parallel with the project timeline. The most significant of those are the following:

- Servers: Some servers have been recently purchased to update the basic infrastructure where most services and software run. The servers purchased and available at the platform are the Dell PowerEdge R740, and its configuration includes:
    - Intel Xeon Gold 5220S CPU with 18 cores at 2.7 GHz
    - 128 GB DDR4 dual channel RAM (2x64 GB)
    - 480 GB SDD x2 and 1 TB HDD x2 for storage
    - 4+1 Ethernet 10 Gbps ports

- Keysight UXM 5G Network Emulation Solution

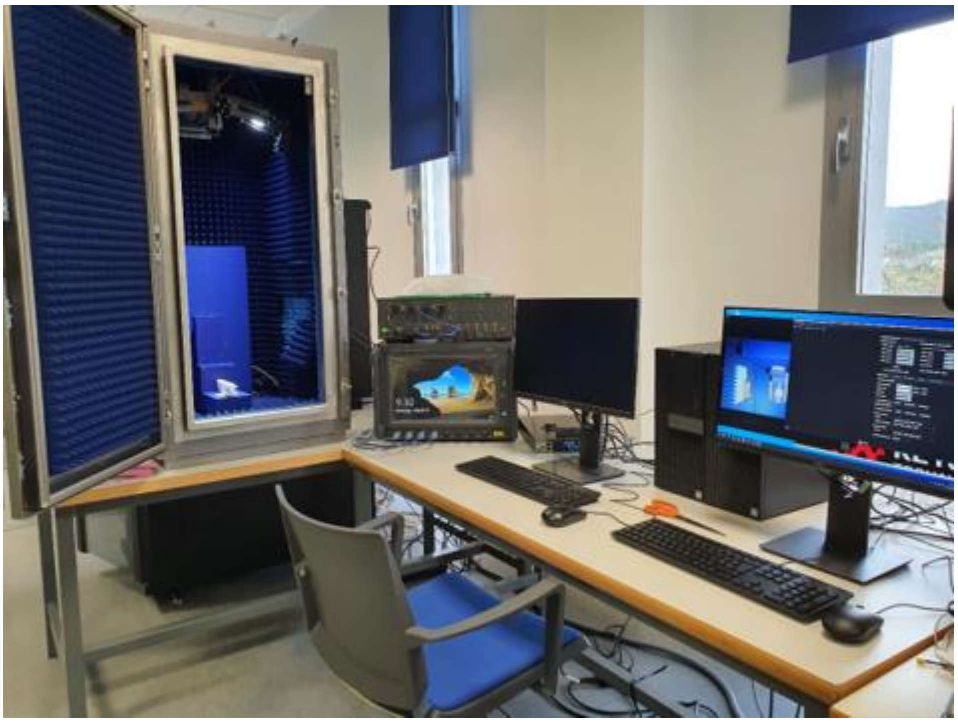


Figure 49: Keysight UXM 5G

The Keysight E7515B UXM 5G wireless test solution [36] is an integrated test platform for 5G network emulation which brings a wide variety of capabilities allowing deep and flexible testing of device RF characteristics, protocol compliance, and functional KPI. It supports the latest 3GPP Release 15 and beyond and works with both NSA and SA 5G modes as well as both FR1 and FR2 frequency bands, in addition to LTE and C-V2X signalling formats.

Some of its specific capabilities include:

- 5G NR 8CC DL, 4CC UL 2x2, with LTE 2CC

  - Wide bandwidth in each RF port
  - Multiple angles of arrival (AoA) test
  - Internal fading for 5G NR and LTE formats
  - Frequency extensions to high IF and millimetre-wave (mmWave) with the use of a common interface unit and remote radio heads (RRH)

  Apart from the E7515B UXM unit, an isolation chamber and two other instruments have also been integrated into the Málaga platform to allow the solution to work with higher frequencies: the M1740A mmWave Transceiver (usually called the Remote Radio Head or RRH) and the E7770A Common Interface Unit (usually called the CIU). Thanks to those additional units mmWave can also be tested with this solution, and even precise DUT positioning for the mmWave beamforming use cases, using the chamber functionality to move and rotate the DUT through remote control of its moving platforms.

- Nokia RAN extension: Nokia radio elements at Malaga's platform includes a huge of capacity scaling, ultra-low latency and huge connectivity extreme capacity. These elements are: AEUB AirScale MAA 8T8R 512AE n257 8W [37], shown in Figure 50, and AirScale Indoor Radio System Small Cell Solution ASiR-pRRH (pico RRH): AWHQB 5G n78 [38], in Figure 51. Some of its specific capabilities include:

  <u>AEUB AirScale MMA 8T8R 512AE n257 8W.</u>
  - 5G Band/Frequency: n257, n261
  - RF Output Power: BRP 60 dBm 2T2R
  - RF Bandwitdh:
    - OBW: 800 MHz
    - IBW: 1400 MHz
  - Carriers: Up to eight 100 MHz 5G NR carriers
  - Synchronization: via AirScale BBU
  - Antenna:
    - 2Tx/2Rx, 2 singles polarized 16x16 arrays
    - Horizontal Coverage Angle +/-45º
    - Vertical Coverage Angle +/-45º
    - Horizontal BW: 6º
    - Vertical BW: 6º

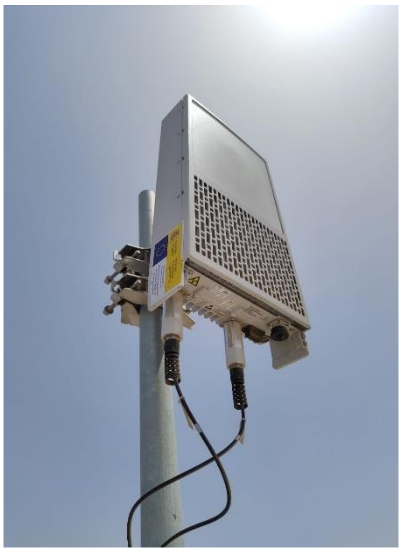


Figure 50: AirScale MAA 8T8R 512AE

AirScale Indoor Radio System Small Cell Solution AsiR-pRRH (pico RRH): AWHQB 5G n78.

- 5G Access: n78 Band 43 Frequency range
- Band Frequency:
  - UL: 3600-3800 MHz
  - DL: 3600-3800 MHz
- RF Output Power: 50mW to 250mW per Tx path
- RF Bandwidth:
  - OBW: 800 MHz
  - IBW: 1400 MHz
- Carrier Bandwidth: 50MHz, 100MHz
- Synchronization: via CPRI timing recovery
- Antenna:
  - Configuration: 4 Tx/4 Rx (per band)
  - Integrated Omni
  - Nominal Gain: -0dBi

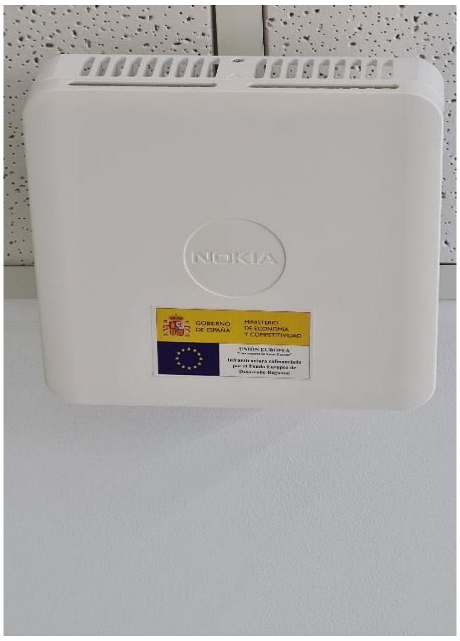


Figure 51: AirScale Small Cell

5.2 Mapping details

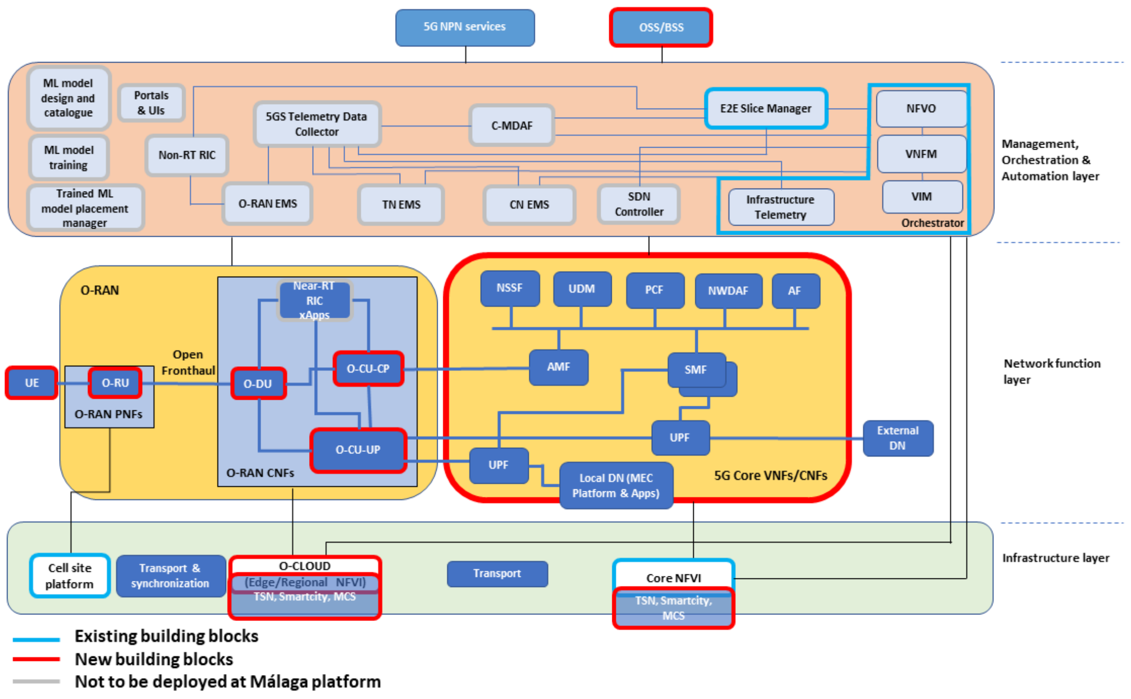


Figure 52: High Level architecture in Málaga

Figure 52 depicts an overview of the high-level architecture with specific mapping for Malaga platform. Elements in the picture have been classified in two categories: Existing building blocks and new building blocks. In addition, elements that are not expected to be implemented in Malaga platform have been highlighted in grey.

5.2.1 Existing building blocks

These are elements provided by other projects (like 5GENESIS) or which were already at Malaga platform before Affordable5G.

E2E Slice Manager: The Katana Slice Manager [39] was introduced as part of the Malaga Platform Release A, and it has been updated to a newer version for Release B. Its focus is to deal with the experiment slices, taking over the command of instantiation of network services in the NFVO. Katana is installed in a dedicated server with network access to the NFVO. Slice Manager for Release B provides the following functionalities:

- VIM user and project management, creating a new user and project for each deployment to improve the isolation (operations supported by OpenStack [40] and OpenNebula [41] VIM types).
- NFVO VIM management, creating a new VIM in the NFVO to complement the above bullet point.
- NS service deployment as part of a new slice, making use of all the items created in the mentioned points. To carry out the previous features, the Slice Manager needs to be preconfigured with information such as the available PoPs, the network services that are going to be deployed and where (PoP), the type of slice, etc

OSM: OSM [32] is an open-source MANO aligned with the ETSI framework for NFV. OSM is an orchestration and management system which manages life-cycle and configuration aspects of the hosted virtual network functions (VNFs) that are deployed on the wide number of supported NFV Infrastructure (NFVI) platforms. These MANO capabilities are critical to implement the sophisticated services expected by the 5G communication systems and utilize the underline management systems and tools. For the purpose of the 5GENESIS Platform, the deployed OSM is of release 6 and is already integrated with the Element management systems and monitoring tools as well as the Virtualisation Infrastructure Managers (Openstack).

Cell site platform:
OpenAirInterface RAN (OAI-RAN) solution [42] provided by Eurecom is an open-source software and hardware platform providing a standard-aligned implementation (3gpp Rel. 10/14) for the LTE UE and eNB. Currently, OAI is being extended to support 5G-NR UE and gNB, as per Rel.15 standards. The Málaga Platform has integrated the OAI 5G-NR UE component, which will be interoperable with the gNB provided by RunEL to perform end-to-end experimentation and KPIs measurement collection. Until the interoperability with RunEL is ready, the platform will use an OAI 5G gNB that has been integrated to allow testing. The currently OAI setup integrated in the Málaga Platform. The protocol stack extensions for both 5G-NR UE and gNB are becoming gradually available throughout the different phases of 5GENESIS, starting from the physical layer (phase 1) and continuing with the rest of the RAN protocol stack (MAC, RLC, RRC, PDCP). The OAI UE can be launched and configured easily through a Command Line Interface (CLI). Based on this CLI, the UE can also be controlled remotely through external software.

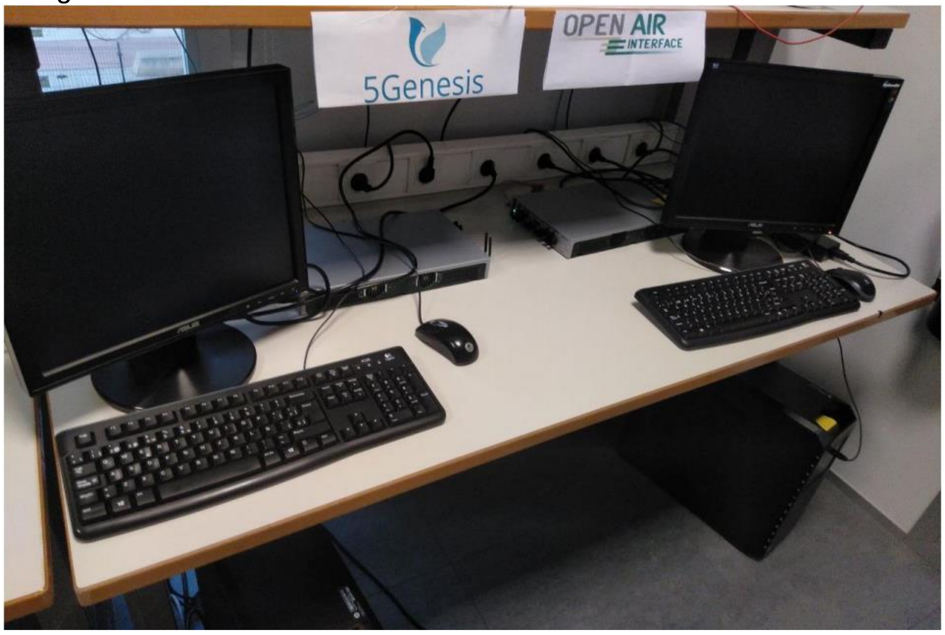


Figure 53: RunEL 5G Infrastructure

RunEL has also provided a 5G Infrastructure to the 5GENESIS Málaga Platform including 5G New Radio (PHY and MAC) already optimized for Ultra Reliable Low Latency Communication (URLLC). The RunEL gNB includes advanced features such as: 2 frequency bands 3.5GHz and 28 GHz, Beam Forming, MIMO, flexible frames, 200MHz BW and more.

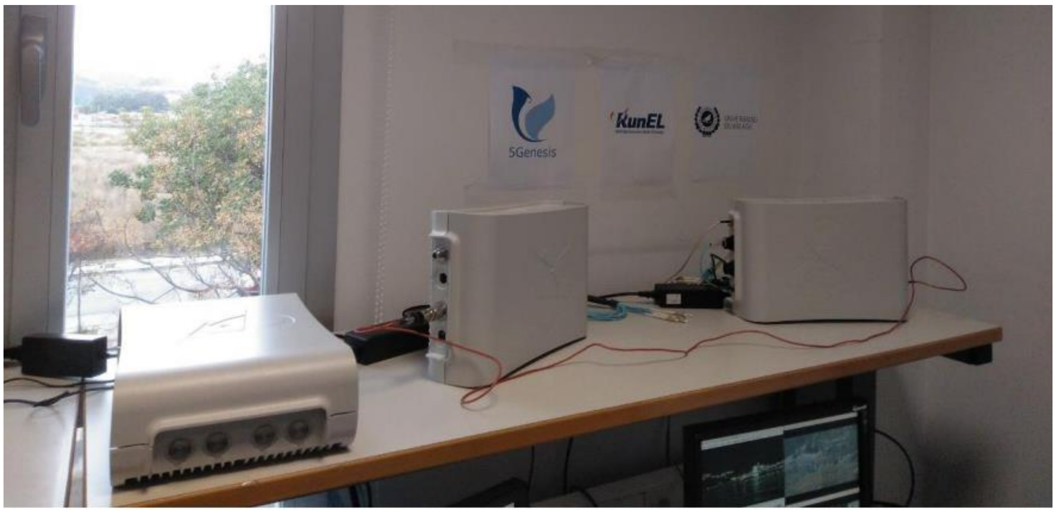


Figure 54: Additional view of RunEL 5G Infrastructure

The RunEL equipment provides a 5G physical layer implementation. The setup includes the main two units which comprise the physical layer: the DRAN (Distributed RAN) unit and the RRH (Remote Radio Head) unit. To enable testing of this units, the setup includes also a UE emulator and software for a basic MAC layer and a video server. The RunEL setup is shown in the following figure (Figure 55):

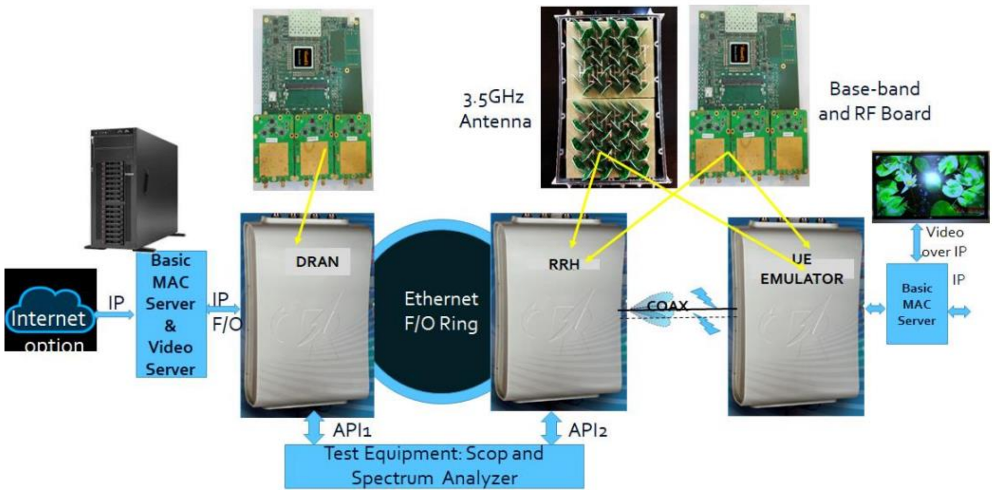


Figure 55: RunEL setup

UMA and other partners from 5GENESIS project have designed the expected outdoor deployments for UMA campus and the city centre to be covered within the 5GENESIS budget. The technology integrated in the platform for the outdoor deployment is the Nokia AirScale Solution, which can be seen in Figure 56 and Figure 57. This solution includes the following components:

- Common equipment
    - Nokia AirScale System Module Indoor. BaseBand Unit, supporting a variety of technologies as GSM, WCDMA, TDD-LTE, FDD-LTE and 5G NR, and a capacity of up to 10 Gbps per system module with up to 96 LTE cells.
- 5G equipment o Nokia Micro RRH 5GC001274. RRH for 5G.
- LTE equipment o Nokia Micro RRH 474147A. RRH for LTE.

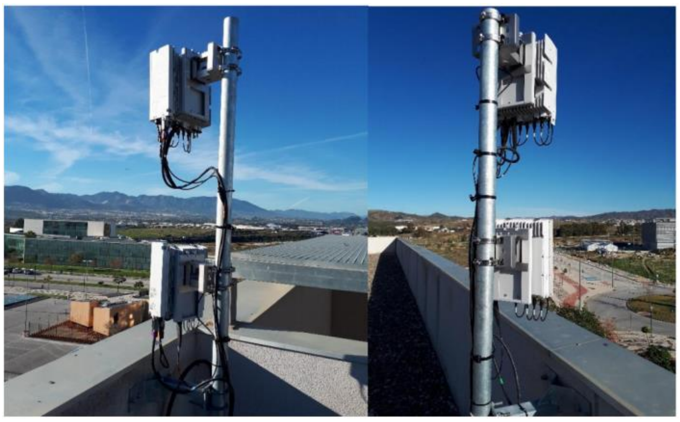


Figure 56: Nokia 5G and 4G micro RRH in Ada Byron building rooftop

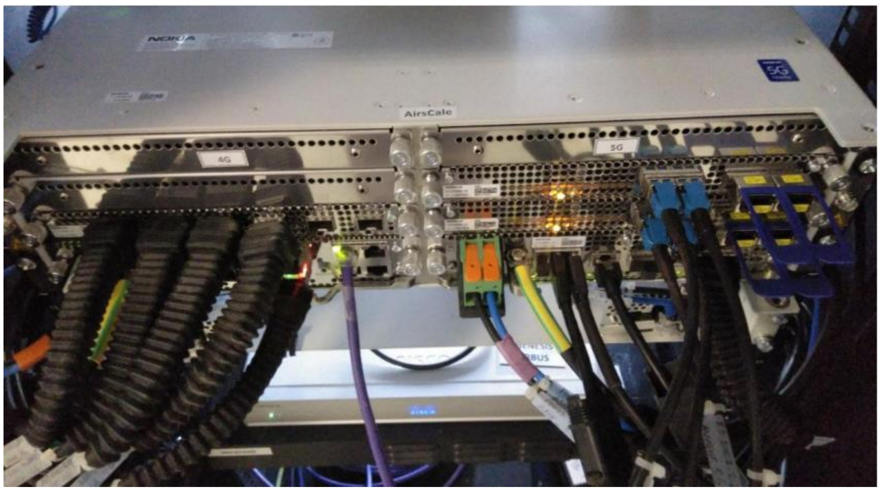


Figure 57: Nokia Airscale BBU at Ada Byron site

Core NFVI:
The Málaga Platform NFVI is managed by two different technologies distributed in two different domains: OpenStack for the Core NFVI and Open Nebula for the Edge infrastructure. In the Main data center, located at the UMA campus, there are three dedicated servers to host and manage the network service instances using OpenStack (Rocky release): the controller, the compute and the storage nodes. In the Edge NFVI, an Open Nebula (v5.8.1) portal and a controller are available. On top of the infrastructure management, there is a single orchestrator handling the NFV deployments in both the Core and the Edge. In this case, the edge orchestrator will not be actually necessary as these functionalities are delegated to the Core NFVO, which has been implemented with OSM Release SIX in Release B. On top of that, we have developed and integrated a wrapper to perform the necessary actions over the NFVO and the VIM, bypassing the security introduced by the components themselves, simplifying the usage to adapt it to 5GENESIS and enhancing some features like the advanced descriptor validation. A simplified network diagram of the above deployment is shown:

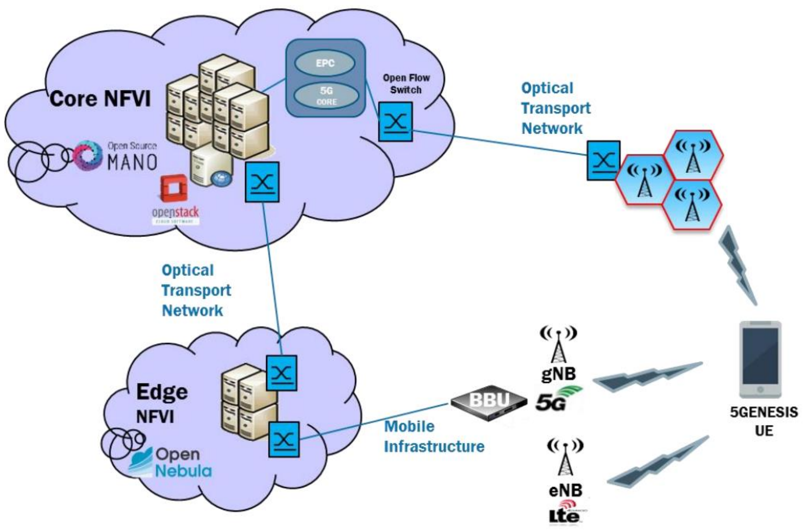


Figure 58: Core NFVI and Edge Composition

5.2.2 New building blocks

These are elements provided by Affordable5G partners to be integrated at Malaga platform.

O-RAN (O-RU, O-DU, O-CU-CP, O-CU-UP):
In Affordable5G project we are using the O-RU provided by RunEL as it has been commented in the subsection before. We have to deploy RunEL elements over the building rooftop (outdoor deployment). O-DU, O-CU-CP and O-CU-UP are also intended to be deployed in the Malaga's platform. Thus, the Affordable5G O-RAN will replace the existing RAN based on commercial equipment described in existing building blocks section. Optical fiber will be the media used for interconnection between these elements (Fronthaul, Midhaul and Backhaul). We can see the current status of Malaga's platform in Figure 59:

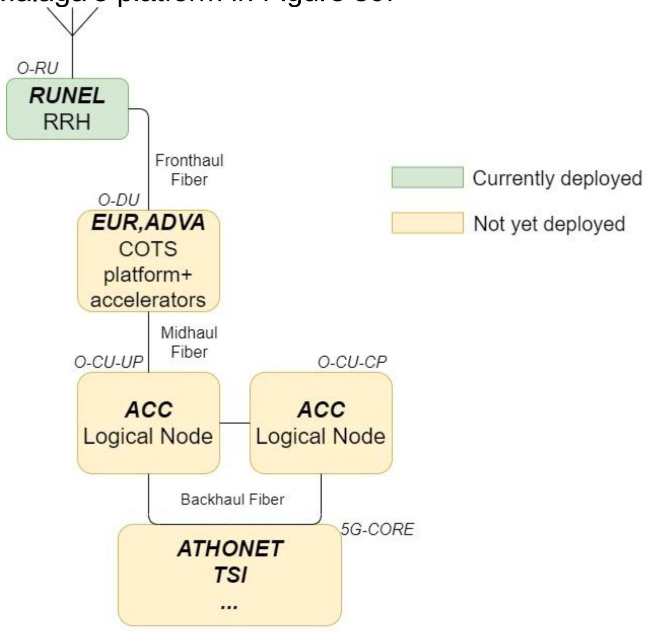


Figure 59: Interconnection O-RAN elements

5GC:
We will use the ATH 5G core. It establishes reliable, secure connectivity to the network for end users and provides access to its services. Athonet will provide UMA with a SA fully virtualized 5GC yet described in the section 2.2 in D2.1. It will have a minimum set of required

functionalities, inherited from the legacy 4G/5G-NSA Software Mobile Core. The complete set of Virtualized Network Functions available will be:

- *AMF*: The Access and Mobility Management Function.
- *SMF*: The Session Management Function.
- *UPF*: The User Plane Function.
- *UDM*: The Unified Data Management.
- *AUSF*: The Authentication Server Function.
- *UDR*: The Unified Data Repository.
- *PCF*: The Policy Control Function.
- *NRF*: The Network Repository Function.
- *NSSF*: The Network Slice Selection Function.

O-CLOUD:
The services described (Smartcity, MSC) will be running in the O-RAN infrastructure described in the different building blocks in section 3, where we have a cloud computing platform using the physical nodes. It allows us to create and host VNFs and have some computing capacity to host and run the services.

TSN:
As described in the 3.9 section, the hardware elements integrated to conform TSN service will be a Relyum TSN setup (2 TSN PCIe cards+1 TSN Bridge); and 4 Intel I210 as end stations. All of it, working with the software elements TSN translators, CNC and CUC.

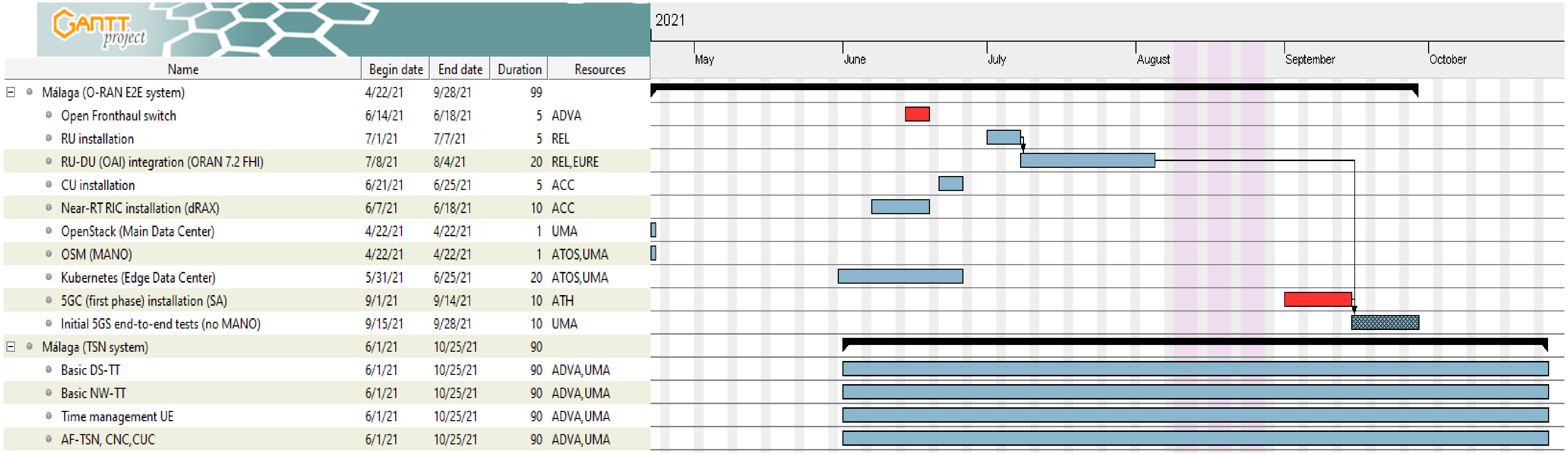
Smartcity:
The Ubiwhere network service to analyse video streams from security cameras in critical points of the city will be integrated as a service in the Malaga platform, being deployed at the network's edge. It is a software solution that could detect in real-time possible dangerous situations.

MSC:
The MSC service will be provided by Nemergent solution. It consists of VNFs/CNFs and Android APP. It is a client/server-based apps, and the VNF has different Virtual Deployments Units (MSC PAS, MSC CAS, MSC LB, MSC DB and MSC xMS). The structure details and specifications are more extensively explained in deliverable 1.2, section 4.2.1. where we can review more information about MSC.

OSS/BSS:
The OpenTap platform will do the functionalities of the OSS/BSS, replacing it. It will do the interface to control and do the proposed testing, managing and monitoring the different components, providing statistics and metrics from the various test cases defined in section 6 below.

5.3 Planning for integration

Similar to the discussion in section 4.3 regarding the planning for integration and testing at the Castelloli platform, the respective timeplan for Málaga is illustrated in Figure 60.



Name	Begin date	End date	Duration	Resources
Málaga (O-RAN E2E system)	4/22/21	9/28/21	99	
Open Fronthaul switch	6/14/21	6/18/21	5	ADVA
RU installation	7/1/21	7/7/21	5	REL
RU-DU (OAI) integration (ORAN 7.2 FHI)	7/8/21	8/4/21	20	REL,EURE
CU installation	6/21/21	6/25/21	5	ACC
Near-RT RIC installation (dRAX)	6/7/21	6/18/21	10	ACC
OpenStack (Main Data Center)	4/22/21	4/22/21	1	UMA
OSM (MANO)	4/22/21	4/22/21	1	ATOS,UMA
Kubernetes (Edge Data Center)	5/31/21	6/25/21	20	ATOS,UMA
5GC (first phase) installation (SA)	9/1/21	9/14/21	10	ATH
Initial 5GS end-to-end tests (no MANO)	9/15/21	9/28/21	10	UMA
Málaga (TSN system)	6/1/21	10/25/21	90	
Basic DS-TT	6/1/21	10/25/21	90	ADVA,UMA
Basic NW-TT	6/1/21	10/25/21	90	ADVA,UMA
Time management UE	6/1/21	10/25/21	90	ADVA,UMA
AF-TSN, CNC,CUC	6/1/21	10/25/21	90	ADVA,UMA

Figure 60: Timeplan for Málaga

6 TEST CASES (AND ACCEPTANCE CRITERIA)

In this section a compilation and grouping of test cases to be performed in the platforms is presented. It is expected to test with them the different building blocks' functionalities, and to have different test cases depending on specific building blocks involved in each test case.

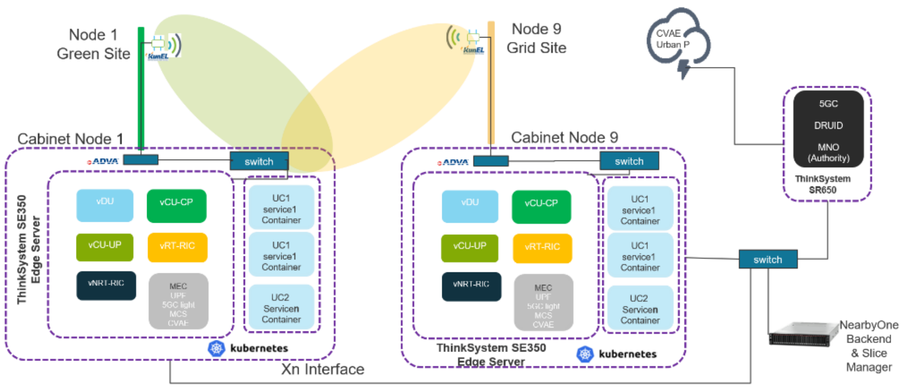
These tests cases aim to test the functionality of the building blocks working together. Thus, the objective is to see if all building blocks have the expected functionality when they are interconnected.

There is a table for every test case defined, based on the template presented in section 3, where we included information about the building blocks involved, the tools used and the verdict of the test. In addition, before each test case is presented, it is explained whether it is a test case specific for one platform or a common one.

The test case exposed in Table 13 is specific for Castellolí platform:

Table 13: Test case int-test-01-01

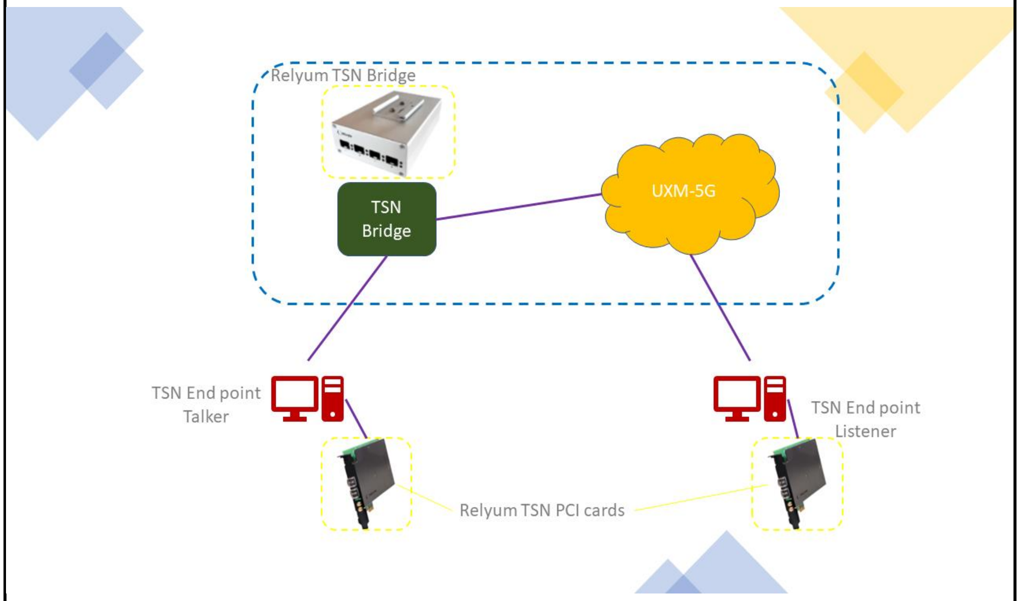
Test case name	**Smartcity Edge Nodes**	**Test Case id**	**int-test-01-01**
Test purpose	Detection and triggering of emergency situations		
Configuration	Elements connected as depicted in figure below.		
Test tool	n/a. Functional testing		
Metric	n/a. Functional testing		
Applicability	O-RAN (O-RU, O-DU, O-CU-CP and O-CU-UP), 5GC		
Pre-test conditions	All elements connected according to configuration. Network orchestrated by MANO.		
Test sequence	Step 1	The real time video stream is processed by the 5GSA Core and detects and emergency situation	5GSA Core reports the event to the Urban platform, which notifies it to the Operator.
	Step 2	The Operator requests real time video feed.	Edge Nodes provides real time video feed passing through the Urban Platform
	Step 2’	The Operator requests event video recording	Edge Nodes provides real time video recording URL stream, and the Urban platform presents the event video recording to the operator.
	Step 3	Operator contacts Civil authorities	End
Test Verdict	It is possible to watch real time video stream. It is possible to watch events video recording.		PASS

Additional Resources	 Node 1 Green Site KunEL Node 9 Grid Site KunEL CVAE Urban P Cabinet Node 1 ADVA switch ThinkSystem SE350 Edge Server vDU vCU-CP vCU-UP vRT-RIC vNRT-RIC MEC UPF 5GC light MCS CVAE UC1 service1 Container UC1 service1 Container UC2 Servicen Container kubernetes Xn Interface Cabinet Node 9 ADVA switch vDU vCU-CP vCU-UP vRT-RIC vNRT-RIC MEC UPF 5GC light MCS CVAE UC1 service1 Container UC1 service1 Container UC2 Servicen Container ThinkSystem SE350 Edge Server kubernetes 5GC DRUID MNO (Authority) ThinkSystem SR650 switch NearbyOne Backend & Slice Manager

The following test case included in Table 14 is exclusive for Málaga platform:

Table 14: Test case int-test-02-01

Test case name	**TSN over UXM 5G**	**Test Case id**	**int-test-02-01**
Test purpose	Functional testing for proof of concept of TSN over 5G using UXM to emulate RAN and 5G Core.		
Configuratio n	Elements connected as depicted in figure below. TSN endpoint (talker) connected to TSN Bridge, which is in turn connected to UXM. UXM must be also connected to TSN endpoint (listener).		
Test tool	Linuxptp, iperf.		
Metric	n/a. Functional testing		
Applicability	TSN endpoints, TSN bridge, UXM 5G		
Pre-test conditions	All elements connected according to configuration. TSN endpoints with linuxptp package and iperf installed.		
Test sequence	Step	Connect the endpoints and configure PTP protocol using ptp4l.	TSN endpoints (talker and listener) should be connected and hardware clocks (PHC) should be synchronized.
	Step	Synchronize in each TSN endpoint the PHC with the system clock using phc2sys.	All clocks should be synchronized.
	Step	Use iperf to check compliance with PTP defined parameters.	
Test Verdict	If connection between TSN endpoints is established according to defined parameters and all clocks are well synchronized.		PASS

Additional Resources	 Relyum TSN Bridge TSN Bridge UXM-5G TSN End point Talker TSN End point Listener Relyum TSN PCI cards

Finally, the test cases included in Table 15, Table 16, Table 17 and Table 18 are common tests for both platforms:

Table 15: Test case int-test-03-01

Test case name	**Affordable O-RAN and 5GC**	**Test Case id**	**int-test-03-01**
Test purpose	Functional testing for O-RAN + ATH 5GC using commercial UE and video / audio streaming server based on VLC.		
Configuration	Elements connected as depicted in figure below.		
Test tool	VLC		
Metric	n/a. Functional testing		
Applicability	O-RAN (O-RU, O-DU, O-CU-CP and O-CU-UP), 5GC		
Pre-test conditions	All elements connected according to configuration. To configure the full network. UE and server with VLC installed.		
Test sequence	Step	Configure a video / audio server and start streaming using VLC.	Video / audio server is ready and streaming the content.
	Step	Connect the UE to video / audio server and play the content using VLC app at UE.	The video / audio is played in the UE.
Test Verdict	If the content streamed by the server is played in the UE.		PASS
Additional Resources			

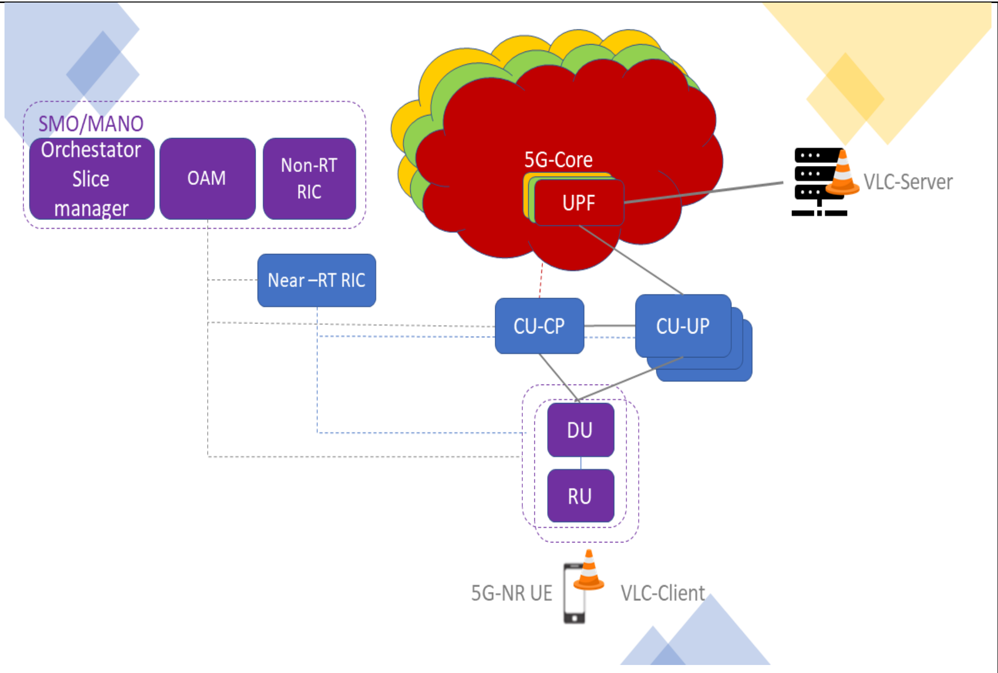
	 SMO/MANO Orchestator Slice manager OAM Non-RT RIC Near –RT RIC 5G-Core UPF VLC-Server CU-CP CU-UP DU RU 5G-NR UE VLC-Client

Table 16: Test case int-test-04-01

Test case name	**Affordable MCS**	**Test Case id**	**int-test-04-01**
Test purpose	Functional testing for NEM MCS using NEM MCS client App on commercial UEs and MCS service deployed.		
Configuration	Service elements deployed and MCS client App provisioned.		
Test tool	MCS protocol itself		
Metric	Functional testing – MCPTT Call and media-floor[1] control		
Applicability	MCS		
Pre-test conditions	To configure the full network. UE with MCS client App installed and MCS service deployed. MCS client App provisioned and opened.		
Test sequence	Step	Configure the MCS service.	MCS service is up & running.
	Step	Register the MCS client App(s) in the MCS service.	The MCS client(s) is/are registered in system and now is/are able to call either private contacts or groups.
	Step	Make MCPTT call	MCPTT call is established in involved MCS clients.

[1] **Media-floor control entity**: A media control resource shared by participants in an MCPTT session, controlled by a state machine to ensure that only one participant can access the media resource at the same time. [43]

	Step	Floor control exchange between call participants.	The MCS client(s) are able to exchange token and talk with token control.
Test Verdict	If the MCPTT call is established and audio can be exchanged from one client to the other(s).		PASS
Additional Resources	n/a		

Table 17: Test case int-test-05-01

Test case name	**Affordable CVAE Network Requirements**	**Test Case id**	**int-test-05-01**
Test purpose	Assess network requirements for the CVAE service		
Configuration	CVAE service deployed and instantiated. Functions correctly chained and working properly.		
Test tool	CVAE internal testing routines		
Metric	Network testing – latency, throughput and packet loss		
Applicability	Full Affordable5G architecture deployed in Málaga or Castellolí		
Pre-test conditions	CVAE service deployed at the core network and at least one edge network. At least one CCTV camera connected via 5GNR to the edge computing.		
Test sequence	Step 1	Enable CVAE testing mode	CVAE service is instantiated and its functions are chained between the core and the edge.
	Step 2	Configure CVAE parameters on CCTV camera and edge functions	The CVAE testing mode is now enabled and test routines are available for execution
	Step 3	Trigger test routine designed to test the E2E network capabilities, from the Camera until the core functions	CVAE service components are now configured and ready to start testing.
	Step 4	Network performance tests terminated	The CVAE service reports the resultant network metrics
Test Verdict	Evaluated network parameters must be within the predefined thresholds.		PASS
Additional Resources	n/a		

Table 18: Test case int-test-05-02

Test case name	**Affordable CVAE Functionality**	**Test Case id**	**int-test-05-02**
Test purpose	Functional testing for CVAE		
Configuration	CVAE service deployed and instantiated. Functions correctly chained and working properly.		
Test tool	CVAE internal testing routines		
Metric	Functional testing – detection of emergency situation		
Applicability	Avail the correct detection of emergency scenarios		
Pre-test conditions	CVAE service deployed at the core network and at least one edge network. At least one CCTV camera connected via 5GNR to the edge computing.		
Test sequence	Step 1	Enable CVAE testing mode	CVAE service is instantiated and its functions are chained between the core and the edge.
	Step 2	Configure CVAE parameters on CCTV camera and edge functions	The CVAE testing mode is now enabled and test routines are available for execution
	Step 3	Trigger test routine designed for assess emergency detection functionality	CVAE service components are now configured and ready to start testing.
	Step 4	Emergency detected and classified	The CVAE service reports an emergency detection and classifies the type of emergency.
Test Verdict	CVAE service must detect an emergency scenario. CVAE service must correctly identify/classify the type of emergency.		PASS
Additional Resources	n/a		

7 CONCLUSIONS

This deliverable focused on how to integrate and test the different blocks from multiple partners that build up the final Affordable5G system. To do this, a methodology to be followed have been defined. This integration methodology has been based in the incremental incorporation of the elements, in such a way that the risks of including new building blocks are limited, until the complete solutions are integrated. On the other hand, with respect to testing, the methodology has been based on the use of OpenTAP, which offers a quick and orderly execution of all the steps to perform the different test cases.

In order to be able to follow this methodology, it was necessary to assess, for each building block, whether they can be accessed through OpenTAP or not. In addition, for these affirmative cases, a table with the interfaces that can be controlled, configured, or monitored with OpenTAP has been provided. After performing this task, the building blocks that can be accessed through OpenTAP are: 5GCORE, all building blocks in O-RAN (O-CU, O-DU, O-RU, Near-RT RIC (dRAX)), AI/ML Framework, Telemetry, TSN over 5G solution from ADVA-REL, TSN over 5G solution from UMA-ATH and both services: Smartcity and MCS.

Another relevant information addressed is the difference between Castellolí and Málaga testbed architectures. This have been done through the representation of the specific mapping for each location, indicating the building blocks that are going to be integrated in each of them. Based on this differentiation, specific test cases arose for each platform, but most test cases are identical. In concrete, these are the four common test cases that have been developed for both platforms: One functional test case for O-RAN + ATH 5GC and three test cases for the services, one for the functional testing of MCS and two additional test cases for the Smartcity.

Finally, this deliverable has a direct connection to future deliverable *D4.2: 5G roll-out and system testing report*, as it will include the upgrade of Castellolí and Málaga platforms with the installation and deployment of enhanced products, developed prototypes and open platforms to carry out complete E2E system tests. The rationale is that the installation and deployment is based on the methodology that has been detailed in this deliverable, which is crucial for execution of these tasks.

REFERENCES

[1] Think Silicon, NEOX^{TM} graphics [Online], https://www.think-silicon.com/neox-graphics, Accessed: August 2021.

[2] RISCV, The RISC-V Instruction Set Manual [Online]. https://riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf, Accessed: August 2021.

[3] Tensorflow, Tensorflow Lite [Online], https://www.tensorflow.org/lite/, Accessed: August 2021.

[4] Affordable5G, D2.1. Hardware elements and usage in Affordable5G solutions [Online], https://www.affordable5g.eu/deliverables/, Accessed: August 2021.

[5] Think Silicon, NEMA® | Bits [Online], https://www.think-silicon.com/?section=2185, Accessed: August 2021.

[6] Xilinx, Xilinx XC706 [Online], https://www.xilinx.com/products/boards-and-kits/ek-z7-zc706-g.html, Accessed: August 2021.

[7] Affordable5G, D3.1. Open platform usage developments [Online], https://www.affordable5g.eu/deliverables/, Accessed: August 2021.

[8] OpenTap [Online], https://doc.opentap.io/, Accessed: August 2021.

[9] 5GENESIS [Online], https://5genesis.eu/, Accessed: July 2021.

[10] UMA OpenTAP SSH Plugin [Online], https://gitlab.com/OpenTAP/Plugins/university-of-malaga/uma-ssh-plugin, Accessed: July 2021.

[11] UMA Android TAP plugin [Online], https://gitlab.com/OpenTAP/Plugins/university-of-malaga/uma-android, Accessed: July 2021.

[12] iPerf speed test tool [Online], https://iperf.fr/, Accessed: July 2021.

[13] Open5Genesis Remote iPerf Agent [Online], https://github.com/5genesis/Remote_iPerf_agent, Accessed: July 2021.

[14] Open5Genesis TAP-plugins [Online], https://github.com/5genesis/TAP-plugins, Accessed: July 2021.

[15] UMA Android iPerf Agent [Online], https://gitlab.com/morse-uma/android-iperf-agent, Accessed: July 2021.

[16] UMA ADB Agents TAP plugin [Online], https://gitlab.com/OpenTAP/Plugins/university-of-malaga/uma-adb-agents, Accessed: July 2021.

[17] TS 38.300, 5G; NR; Overall description; Stage-2 (3GPP TS 38.300 version 15.8.0 Release 15) [Online], https://www.etsi.org/deliver/etsi_ts/138300_138399/138300/15.08.00_60/ts_138300v150800p.pdf, Accessed: July 2021.

[18] O-RAN Alliance [Online], https://www.o-ran.org/, Accessed: July 2021.

[19] O-RAN Alliance, O-RAN.WG6.O2-GA&P-v01.00 [Online], https://www.o-ran.org/specifications, Accessed: July 2021.

[20] O-RAN Alliance, O-RAN.WG1.O1-Interface.0-v04.00 [Online], https://www.o-ran.org/specifications, Accessed: July 2021.

[21] Helm charts [Online], https://helm.sh/, Accessed: July 2021.

[22] O-RAN Alliance, O-RAN.WG6.O2-GA&P-v01.00 [Online], https://www.o-ran.org/specifications, Accessed: July 2021.

[23] O-RAN Alliance, O-RAN.WG5.MP.0-v01.00 [Online], https://www.o-ran.org/specifications, Accessed: July 2021.

[24] RunEL, Sparq-2020 Chipset Series for 5G Applications [Online], https://www.runel.net/products, Accessed: August 2021.

[25] O-RAN Alliance, O-RAN.WG3.RICARCH-v02.00 [Online], https://www.o-ran.org/specifications, Accessed: July 2021.

[26] Tensorflow Extended [Online], https://www.tensorflow.org/tfx, Accessed: July 2021.

[27] Apache Airflow Python client [Online], https://github.com/apache/airflow-client-python/tree/master/airflow_client, Accessed: July 2021.
[28] Apache Airflow CLI [Online], https://airflow.apache.org/docs/apache-airflow/stable/cli-and-env-variables-ref.html, Accessed: July 2021.
[29] MCOP / Mission Critical Open Platform [Online], https://www.mcopenplatform.org/, Accessed : August 2021.
[30] RESTful API | TFX | TensorFlow [Online], https://www.tensorflow.org/tfx/serving/api_rest, Accessed: July 2021.
[31] gRPC API | TFX | TensorFlow [Online], https://github.com/tensorflow/serving/tree/master/tensorflow_serving/apis, Accessed: July 2021
[32] Open Source MANO [Online], https://osm.etsi.org/, Accessed: August 2021.
[33] Affordable5G, D1.2. Affordable5G building blocks fitting in 5G system architecture [Online], https://www.affordable5g.eu/deliverables/, Accessed: August 2021.
[34] Relyum TSN products [Online], https://www.relyum.com/web/time-sensitive-networking-products/, Accessed: July 2021.
[35] U-blox, U-blox EVK-R5 [Online], https://www.u-blox.com/en/product/evk-r5, Accessed: August 2021.
[36] Keysight Technologies, E7515B UXM 5G Wireless Test Platform [Online], https://www.keysight.com/es/en/product/E7515B/uxm-5g-wireless-test-platform.html, Accessed: July 2021.
[37] Nokia, AirScale mmWave Radio [Online], https://www.nokia.com/networks/radio-access/airscale/mmwave-radio/, Accessed: July 2021.
[38] Nokia, AirScale indoor Radio [Online], https://www.nokia.com/networks/radio-access/airscale/indoor-radio/, Accessed: July 2021.
[39] Katana Slice Manager [Online], https://github.com/medianetlab/katana-slice_manager, Accessed: August 2021.
[40] OpenStack [Online], https://www.openstack.org/, Accessed: August 2021.
[41] OpenNebula [Online], https://opennebula.io/, Accessed: August 2021.
[42] Open Air Interface, OAI 5G RAN Project Group [Online], https://openairinterface.org/oai-5g-ran-project/, Accessed: August 2021.
[43] 3GPP, 3GPP TS 24.379 V17.3.1 (2021-06) [Online], https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2953, Accessed: August 2021.
[44] 3GPP, Study on channel model for frequencies from 0.5 to 100 GHz. Technical report (TR) 38.901, 3rd Generation Partnership Project (3GPP), 2019
[45] A. Giannopoulos, S. Spantideas, C. Tsinos and P. Trakadas, "Power Control in 5G Heterogeneous Cells Considering User Demands Using Deep Reinforcement Learning", In IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, pp. 95-105, June 2021.
[46] A. Giannopoulos et al., "WIP: Demand-Driven Power Allocation in Wireless Networks with Deep Q-Learning", In 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 248-251, June 2021.

APPENDIX

In this appendix, we consider the preliminary implementation of a practical optimization algorithm in the form of an xAPP. The algorithm is based on a Deep Reinforcement Learning (DRL) based approach, according to which the O-RAN is optimized in terms of experienced throughput. Specifically, given the UE measurement reports (collected from the O-RAN), the algorithm aims to maximize the network throughput by providing a power allocation scheme for all Physical Resource Blocks (PRBs) of all active RUs. To that end, a DRL agent is trained on simulated network measurements obtained from mobile users inside a three-cell area.

The implementation framework starts with a simulation environment in order to (i) train the DRL agent in realistic and 5G-compliant network conditions and (ii) test and validate the deployed algorithm by using the simulated data for model inference purposes. To concretely describe the architecture of this scenario, Figure 61 illustrates the dRAX building blocks, especially focused on the support of xAPPs.

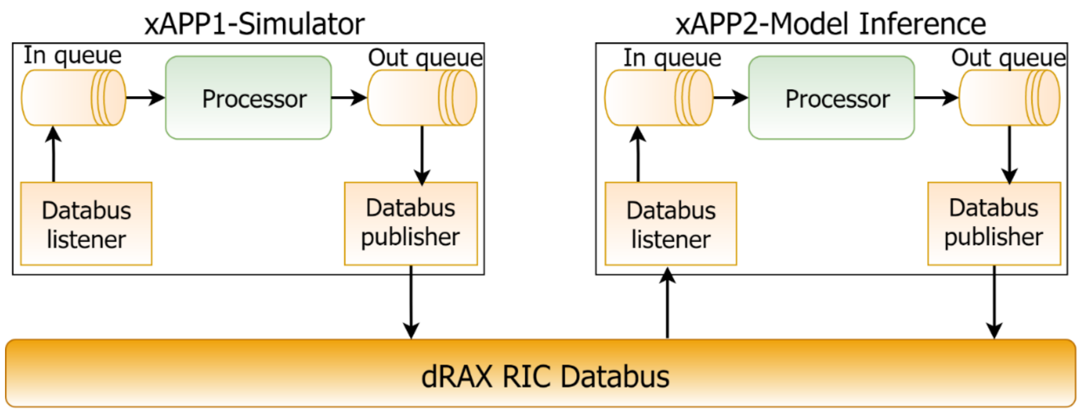


Figure 61: General diagram of the deployed dRAX xAPPs for supporting maximal throughput-targeted power control

As shown in the left part of Figure 61, a 5G network simulator is built in the form of xAPP to provide O-RAN measurements. Figure 62 illustrates the internal functionality of xAPP1. A network area consisting of three macro-cells is considered, following the specification of UMa cells (Urban Macrocells) detailed in [44]. Each RU has 12 available PRBs to transmit data using the OFDM modulation scheme. Inside the network area, a set of mobile users is established, with each individual UE requesting a specific service (expressed in Mbps). At each time instance, each UE follows a random-walk model. The algorithm supports time-varying number of users, given that UEs may enter or exit the network area, according to their trajectories. In addition, the network simulator is aware of possible handovers, since a UE may be served from different RUs depending on which is the best server for a given time point. The key functionality of the simulator includes the interference calculations for each UE by not only considering the channel losses but also the accumulated interference caused by the non-servers. Specifically, the signal-to-interference-plus-noise ratio is computed for each pair of associated UE-RU, taking into account the 5G-compliant UMa channel models [44]. Furthermore, the association scheme is implemented according to the maximum-throughput criterion, meaning that each UE occupies the PRB from which it receives the best throughput. In summary, the algorithm produces the following output UE measurement reports:

- **Received Strength Signal Indicator (RSSI):** measures the average total received power observed only in OFDM symbols containing reference symbols in the measurement bandwidth over 12 resource blocks.
- **Reference Signal Received Power (RSRP):** RSRP is an RSSI type of measurement, proportional to the power of the LTE Reference Signals spread over the full bandwidth and narrowband.

- **Reference Signal Received Quality (RSRQ):** Quality considering also RSSI and the number of used Resource Blocks measured over the same bandwidth. RSRQ is a C/I type of measurement. The RSRQ measurement provides additional information when RSRP is not sufficient to make a reliable handover or cell reselection decision.
- **Channel Quality Indicator (CQI):** ranges from 1-15 depending on the quality of the received signal and the experienced throughput.
- **Cell ID:** a bit indicating the serving cell according to maximum-throughput criterion
- **PRB ID:** a bit indicating the associated PRB of the serving cell according to maximum-throughput criterion
- **Throughput:** a value in Mbps reflecting the experienced throughput from the associated Cell-PRB.

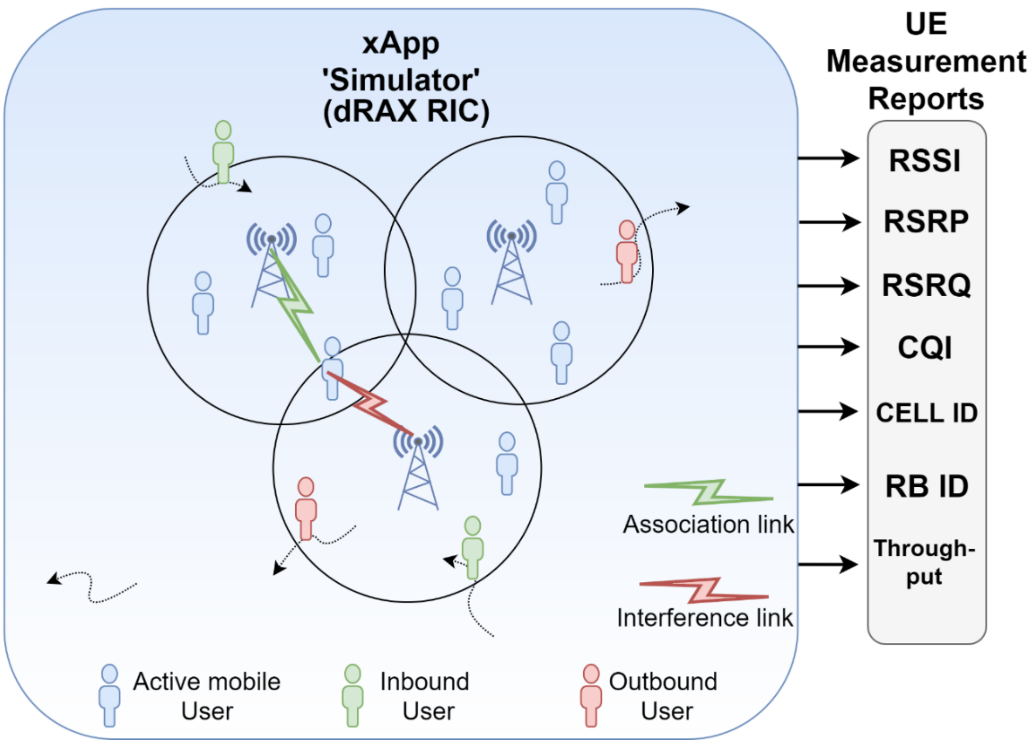


Figure 62: Internal functionality of xAPP1. The simulator supports varying number of mobile users, as well as it uses 5G UMi channel models for the channel propagation losses

At each time point, a UE measurement report vector is published in the dRAX RIC Databus through the Databus publisher. This vector is then available to the Databus listener of the xAPP2 for model inference purposes. Before describing the functionality of the xAPP2, the network model and the training approach of the DRL algorithm is presented.

Network Model: An urban 5G network area is considered to be deployed, consisting of 3 partially overlapped micro-cells (UMi). Each cell k is covered by the respective transmitter k (Tx k, $\forall k = 1,2,\dots,K$). According to the selected operational 5G band and PRB segmentation (5G numerology), each cell has N available PRBs for physical-layer transmissions. A complete frequency reuse scheme across cells is also assumed. The system is controlled by a centralized cognitive controller, which targets at maximizing the network-wide throughput. Each UE ($u = 1,2,\dots,U$) requested a throughput-specific service s defined by the service level agreement (SLA) profile of the available services ($s = 1,2,\dots,S$). Each service corresponds to a particular throughput demand in order to ensure adequate QoS. Therefore, a demand vector D, with respective elements d_i ($i = 1,2,\dots,U$), is adapted to notify the requested service class of user u, expressed in terms of throughput (Mbps). Each single user u may be associated with a PRB n of a particular cell k, thus defining the association matrix A ($A_{k,n,u} = 1$ when the association occurs, or 0 otherwise). The power level of cell k over PRB n is denoted as $P_{n,k}$.

Moreover, a sum-power constraint is established for each cell due to power budget limitations. To account for signaling/sleeping mode operations, a PRB-specific minimum power level is also defined. Noteworthy, we assume that (i) each user is connected to a single PRB, (ii) the cell capacity is upper-bounded by the number of available PRBs and (iii) each cell can cover multiple users (at most N users). The computation of the reachable transmission rate R of the link between Tx k and user u over PRB n is based on the Shannon formula.

Training the DRL agent: In general, RL enables an agent interacting with an environment, and getting feedback loops between the learning system and its experiences, in terms of rewards or punishments. The conventional form of RL is the tabular Q-learning method, according to which the RL agents take advantage of the so-called Q-table to become near-optimal predictors of beneficial actions [45], [46]. During the learning process, the agent records its past experiences by continuously filling-and-updating the Q-table. An immediate extension of the tabular Q-learning is to utilize a neural network (DQL) as Q-function approximator, instead of using a memory-inefficient array structure. In principle, the condition of the environment is acknowledged to the DRL agent through the system state $s \in S$ (state space). The agent can then interact with the environment by performing an action $a \in A$ (action space). The learning process loop is completed with the received reward r, involving the feedback (positive, negative or none) of the performed action, as well as the new state of the environment. The concept of the learning method is to train the agent to gradually prefer the actions that return the most profitable rewards.

The key concept of DQL is the utilization of two function approximators (i.e., neural networks) to estimate the current best action (Q-network) and to predict the next best action (target Q–network). The Deep Learning part of the DQL is basically a regression problem in which the objective is to minimize a loss function. This function equals to the difference between the outputs of the Q- (considered as the predicted values) and target Q-networks (considered as the actual values). As a result of the training process, the weights of the Q-network are properly adjusted so as to provide predictions about which action to take from a given state (inference phase).

The implemented algorithm uses the user-specific triplet of (Cell ID, PRB ID, CQI) as the state vector, whereas the action vector is comprised of the suggested power levels [45]. The reward received at time t is equal to the difference between the current and the previous system throughput.

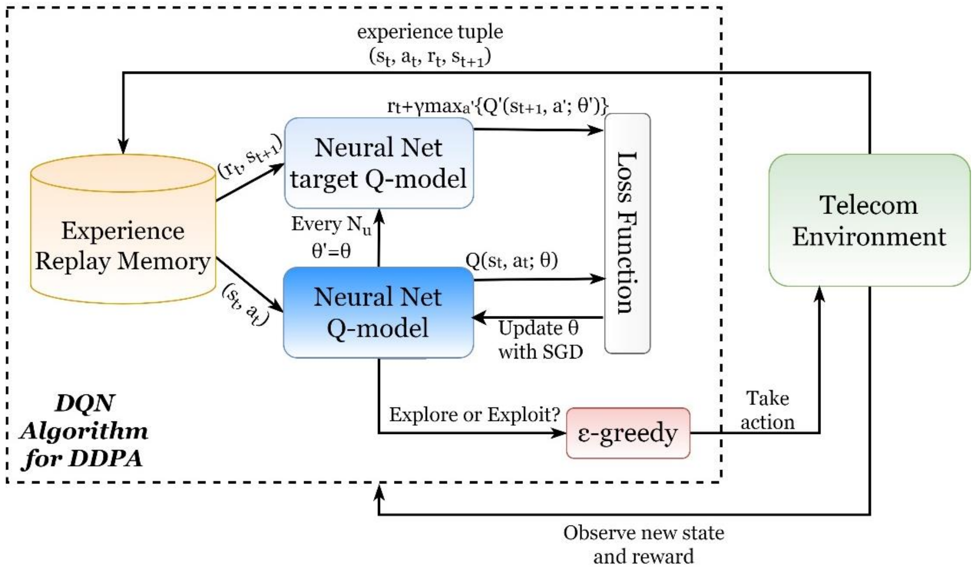


Figure 63: DRL architecture for training the power control xAPP

Based on the architecture of Figure 63, the progression of the training phase can be described in the following steps:

- **Step 1:** The cognitive controller associates each user with a specific RU/RB pair based on the maximum throughput criterion. Each RU/RB is initialized to transmit with random power level before the agent starts to explore the environment.
- **Step 2:** An RB is selected for each RU and its power is regulated depending on the operational mode of the algorithm: in exploration phase, an RB is randomly selected and its power is either increased, decreased or kept fixed, whereas in exploitation phase the action is estimated by the Q-network.
- **Step 3:** The environment provides feedback to the DRL agent regarding the performed action (immediate reward) and the next state. This procedure involves the update of RU/RB association and calculation of the allocated throughput for each user with respect to the new power levels.
- **Step 4:** The system stores the experience tuple (st, at, rt+1, st+1) into the replay memory. In case that the memory is full, the least recently used tuple is replaced. Noteworthy, the memory is initially filled with 1000 experience tuples corresponding to random actions.
- **Step 5:** A batch of experience tuples NB is randomly selected from the memory. The current state st batch elements are forward-passed through the Q-network to predict the Q-values of all actions. The weights of the Q-network neurons are adjusted through the back-propagation method (Stochastic Gradient Descent).
- **Step 6:** Every Nu steps, the weights of the Q-network model are cloned to the target Q-network model.
- **Step 7:** The agent reduces the value of ε (linear decay) to get closer to the exploitation mode and repeats steps 1-6 until convergence.

In this study, the training of the ML model was done offline using Python, Keras and Tensorflow. Exploitation of the capabilities of the AI/ML framework (presented in section 3.4) for model training will be investigated at a later stage.

Having pre-trained the DRL agent, Figure 64 shows the functionality of xAPP2. The key blocks of the xAPP2 correspond to the grey box of Figure 64, where the centralized controller firstly collects the measurement reports from the dRAX RIC Databus (through the Databus listener) and then combine them into a global state vector. Finally, the pre-trained agent is inferred to provide the global power allocation vector (through the Databus publisher).

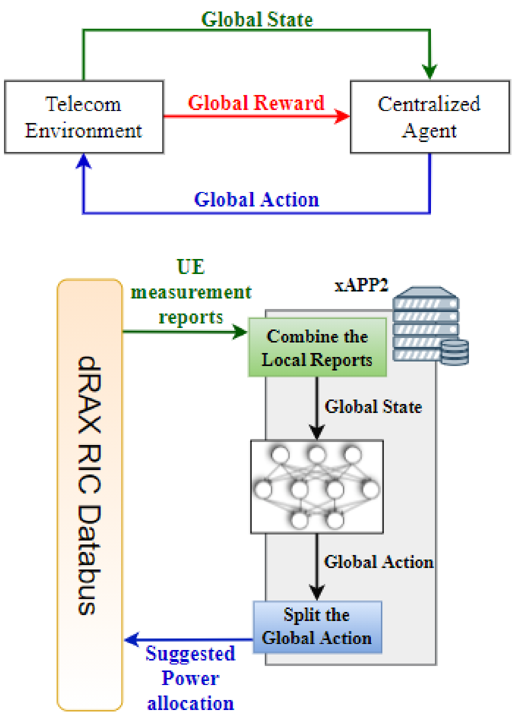
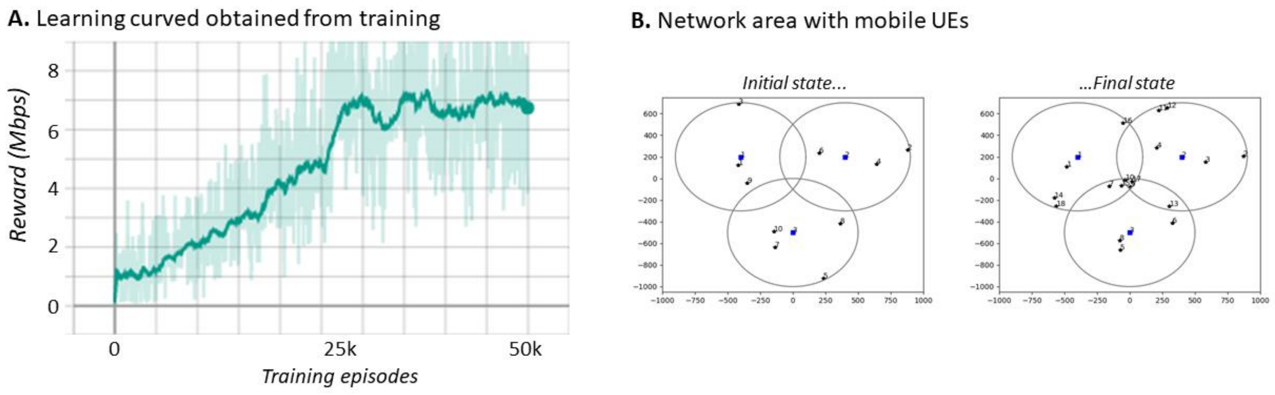
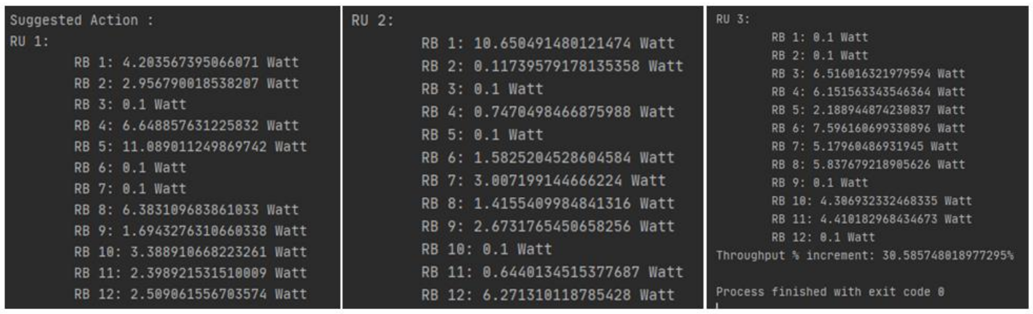


Figure 64: Internal functionality of xAPP2. The pre-trained model is inferred with the data collected from dRAX Databus and publishes back the suggested power allocation scheme in the Databus

Sample results: Both xAPPs have been integrated and implemented following the dRAX technical documentation and the respective pre-built functions. Figure 65 depicts some sample results regarding the APP1 and xAPP2 implementation.



C. Example of a suggested action



```
Suggested Action :
RU 1:
        RB 1: 4.203567395066071 Watt
        RB 2: 2.956790018538207 Watt
        RB 3: 0.1 Watt
        RB 4: 6.648857631225832 Watt
        RB 5: 11.089011249869742 Watt
        RB 6: 0.1 Watt
        RB 7: 0.1 Watt
        RB 8: 6.383109683861033 Watt
        RB 9: 1.6943276310660338 Watt
        RB 10: 3.388910668223261 Watt
        RB 11: 2.398921531510009 Watt
        RB 12: 2.509061556703574 Watt
```

```
RU 2:
        RB 1: 10.650491480121474 Watt
        RB 2: 0.11739579178135358 Watt
        RB 3: 0.1 Watt
        RB 4: 0.7470498466875988 Watt
        RB 5: 0.1 Watt
        RB 6: 1.5825204528604584 Watt
        RB 7: 3.007199144666224 Watt
        RB 8: 1.4155409984841316 Watt
        RB 9: 2.6731765450658256 Watt
        RB 10: 0.1 Watt
        RB 11: 0.6440134515377687 Watt
        RB 12: 6.271310118785428 Watt
```

```
RU 3:
        RB 1: 0.1 Watt
        RB 2: 0.1 Watt
        RB 3: 6.516016321979594 Watt
        RB 4: 6.151563343546364 Watt
        RB 5: 2.188944874230837 Watt
        RB 6: 7.596160699330896 Watt
        RB 7: 5.17960486931945 Watt
        RB 8: 5.837679218905626 Watt
        RB 9: 0.1 Watt
        RB 10: 4.306932332468335 Watt
        RB 11: 4.410182968434673 Watt
        RB 12: 0.1 Watt
Throughput % increment: 30.585748018977295%

Process finished with exit code 0
```

*Figure 65: Sample results of the implemented demo. **A.** Learning curve showing the gradual throughput increment of the DRL algorithm. **B.** The network simulator outcomes (initial and final network state) for a 5-min simulation. **C.** The suggested action provided by the pre-trained agent for a given network instance*