EXPERIMENTATION AND VALIDATION OPENNESS FOR LONGTERM EVOLUTION OF VERTICAL INDUSTRIES IN 5G ERA AND BEYOND

[H2020 - Grant Agreement No.101016608]

Deliverable D5.4

System level evaluation and KPI analysis (Final Version)

Editor Almudena Díaz (UMA)

Contributors TID, UMA, NCSRD, INTRA, COS, INF, MAG, ATOS, LNV, IEA/ENVOLVE, UPV, GMI, ININ, CAF, IQB, FOGUS, 8BELLS, PAL, ZORTENET, IMM, UMS

Version 1.4

Date November 10th, 2023

Distribution PUBLIC (PU)

DISCLAIMER

REVISION HISTORY

Revision	Date	Responsible	Comment
0.1	*June 20, 2023*	*A. Díaz Zayas (UMA)*	*Table of Content*
0.2	*September 5, 2023*	*A. Díaz Zayas (UMA)*	*UMA contributions*
0.3	*September 26, 2023*	*A. Díaz Zayas (UMA)*	*UMA contributions*
0.4	*October 9, 2023*	*A.Díaz Zayas (UMA)*	*Demokritos contributors*
0.5	*October 18, 2023*	*A. Díaz Zayas (UMA)*	*All contributors*
1.0	*October 24, 2023*	*A. Díaz Zayas (UMA)*	*All contributors*
1.2	*November 3, 2023*	*A. Díaz Zayas (UMA)*	*Minor changes*
1.3	*November 9, 2023*	*A. Díaz Zayas (UMA)*	*Minor changes*
1.4	*November 10, 2023*	*A. Díaz Zayas (UMA)*	*Final version*

LIST OF AUTHORS

Partner ACRONYM	***Partner FULL NAME***	***Name & Surname***
TID	*TELEFONICA INVESTIGACIÓN Y DESARROLLO*	*Javier García* *David Artuñedo* *Jorge Moratinos*
UMA	*UNIVERSITY OF MÁLAGA*	*Almudena Diaz* *Bruno García* *Francisco Luque*
NCSRD	*NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”*	*H. Koumaras* *G. Makropoulos* *D. Fragkos* *A. Gogos*
INTRA	*Netcompany Intrasoft*	*Angela Dimitriou*
COS	COSMOTE Mobile Telecommunications S.A	*F. Setaki* *I. Mesogiti*

GLOSSARY

Abbreviations/Acronym	Description
3GPP	3^{rd} Generation Partnership Project
5G NR	5G New Radio
AF	Application Function
AMF	Access and Mobility Management Function
API	Application Programming Interface
CAPIF	Common API Framework
CI/CD	Continuous Integration / Continuous deployment
COTS	Commercial off-the-shelf
DS	Device Side
E2E	End to End
EC	European Commission
ELCM	Experiment Life-Cycle Manager
EN_DC	E-UTRAN New Radio – Dual Connectivity
EPC	Evolved Packet Core
ETSI	European Telecommunications Standards Institute
FOF	Factories of the Future
gNodeB / gNB	Next Generation (5G) Base Station
GPS	Global Positioning System
GUI	Graphical User Interface
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
KPI	Key Performance Indicator
LTE	Long Term Evolution
MANO	Management and Orchestration
MIMO	Multiple-Input Multiple-Output
MCS	Modulation Coding Scheme
NEF	Network Exposure Function
NS	Network Service
NSA	Non-Standalone
NSD	Network Service Descriptor
NSI	Network Slice Instance
NST	Network Slice Template
NW-TT	Network-Side TSN Translator
OFDM	Orthogonal Frequency Division Multiplexing
OSM	Open-Source MANO
P4	Programming Protocol-independent Packet Processors
PCF	Point Coordination Function
PCP	Priority Code Point
PDCP	Packet Data Convergence Protocol (PDCP)
PRB	Physical Resource Block
QAM	Quadrature Amplitude Modulation

QoS	*Quality of Service*
RAN	*Radio Access Network*
RRH	*Remote Radio Head*
RTT	*Round-Trip Time*
SA	*Standalone*
SMF	*Session Management Function*
SUT	*System Under Test*
TCP	*Transmission Control Protocol*
TDD	*Time Division Duplex*
TSN	*Time-Sensitive Networking*
TT	*TSN Translator*
UDP	*User Datagram Protocol*
UE	*User Equipment*
UPF	*User Plane Function*
URLLC	*Ultra-Reliable Low-Latency Communication*
VM	*Virtual Machine*
VIM	*Virtual Infrastructure Manager*
VNF	*Virtual Network Function*
VNFD	*Virtual Network Function Descriptor*

EXECUTIVE SUMMARY

The purpose of this document is to present the final results of the system-level evaluations performed on the EVOLVED-5G platforms during the project's second phase. The document also includes the results of the final verification tests and the final analysis of the performance of the software tools developed within the context of the project, namely the CAPIF Tool and NEF emulator.

The deliverable is the result of the work carried out in Tasks 5.1 of the EVOLVED-5G project, where the main goal is to apply the verification and validation methodology in order to test and quantify a set of Key Performance Indicators (KPIs) in order to assess the capabilities of the EVOLVED-5G infrastructure and software components.

Based on the above, the main contribution of this deliverable is to describe the results obtained during the final validation of the platforms (Malaga, Athens and Cosmote) and the NEF and CAPIF software components. This deliverable includes:

- Results of the final performance assessment of the platforms.
- Final verification and validation of the CAPIF Tool and NEF Emulator.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1. Document Purpose

The main goal of this document, titled *"System level evaluation and KPI analysis (Final version)"* is to present the results obtained during the final evaluation of the EVOLVED-5G platforms, Athens and Málaga. These measurements reflect the improvements stemming from the updates performed to the platforms during the last part of the project (M21-M33) and have been realized through the experimentation methodology defined in Work Package 2 as well as by utilizing tools developed in Work Packages 3 and 4.

1.2. Document Structure

The document is divided into two main sections:

- **Section 2. FINAL Platform Assessment:** This section is divided into three sub-sections:
  - Section 2.1: The EVOLVED-5G Experimentation Methodology presents a summary of the experimentation methodology followed by the EVOLVED-5G project. According to the methodology, new test cases has been elaborated for evaluating the performance of the platforms. The new test cases are included in the Annexes.
  - Section 2.2: The EVOLVED-5G Platforms presents a complete and updated description of the EVOLVED-5G infrastructure platforms (Malaga and Athens). Athens platform has been extended with a new site, while the Malaga platform has been updated with two mmWaves antennas.
  - Section 2.3: Final Tests and Results summarizes the results obtained during the final evaluation of the platforms. More detailed TSN results are provided in this deliverable.
- **Section 3. Component-Level Evaluation:** Section 3 is devoted to the functional and performance evaluation of the CAPIF Tool and NEF Emulator as software components developed within the context of EVOLVED-5G. The new evaluation presented in this deliverable covers the final CAPIF and NEF APIs versions.

1.3. Target Audience

The release of the deliverable is public, intending to showcase results and status of the platforms and software components of EVOLVED-5G. From specific to broader, different target audiences for D5.4 are identified below:

- **Project Consortium**: To assets the final version of the EVOLVED-5G platforms and the performance of the of the latest versions of the NEF and CAPIF APIs.
- **Industry 4.0/Industry 4.0 developers, FoF (Factories of the Future) and other vertical industries and groups:** To showcase the performance and available features in the EVOLVED-5G platforms, which may raise awareness and interest in other industrial partners in the project achievements.
- **The scientific audience, general public and the funding EC Organization**: The scientific audience can get access to the performance results measured in three separate sites that form the two EVOLVED-5G platforms, which can be used as a baseline for future investigation. This deliverable also documents the work carried out by the Project Consortium and justifies the effort reported in the corresponding activities.

2 FINAL PLATFORM ASSESSMENT

2.1 THE EVOLVED-5G EXPERIMENTATION METHODOLOGY

The EVOLVED-5G experimentation methodology is an adaptation of the methodology defined in the 5Genesis project 0, which is based on the definition of Test Cases. Test Cases follow the template shown in Annex 6.1, which includes:

- A short description of the test.
- A listing of any necessary pre-conditions and assumptions that need to be verified before the test execution.
- A description of the target KPI, including any measurement methods or calculations required for obtaining it, and
- The sequence of steps to follow, either manually or automatically, during the test execution. A more complete description of the Test Cases and the methodology can be found in Deliverable D2.2 [2], section 6.4.

Once defined, Test Cases are implemented in the particular testing environment where the tests will be conducted. The process includes any preparation needed for meeting the Test Case pre-conditions (such as installing and configuring any hardware or software requirement), the implementation of any additional functionality required as indicated by each step in the test sequence (especially in the case of automated tests) or any partial testing required to ensure that the Test Case can be correctly executed in the testing environment.

2.2 THE EVOLVED-5G PLATFORMS

The EVOLVED-5G project makes use of two different platforms located in Athens (composed by two sites: NCSR Demokritos and Cosmote) and Málaga. The two platforms provide 5G capabilities and cloud infrastructures where Open5Genesis framework for the coordination of the experiments is deployed.
The two platforms provide support for the execution of the Validation and Certification processes, by making available their containerization environments for the deployment of the Network Applications, as well as a real 5G network that Verticals can use for the execution of additional tests more related to the specific functionality of each particular Network App.

2.2.1 The Athens Platform

The Athens platform is comprised of two testbeds, NCSRD and COSMOTE, which are interconnected through a 10G direct fiber link. For platform assessment, the two sites act as independent full 5G SA solutions that are evaluated using the Open5Genesis experimentation framework, which dictates the lifecycle of the experiments. As shown in Figure 1, Open5Genesis is hosted at NCSRD's premises and manages and orchestrates all the experiments. The first 5G SA network is based on the ATHONET 5G SA Core and ERICSSON BBU/RRU/RAN which is deployed at the COSMOTE campus. The second 5G SA network is deployed at the NCSRD campus and is based on the Amarisoft 5G solution.

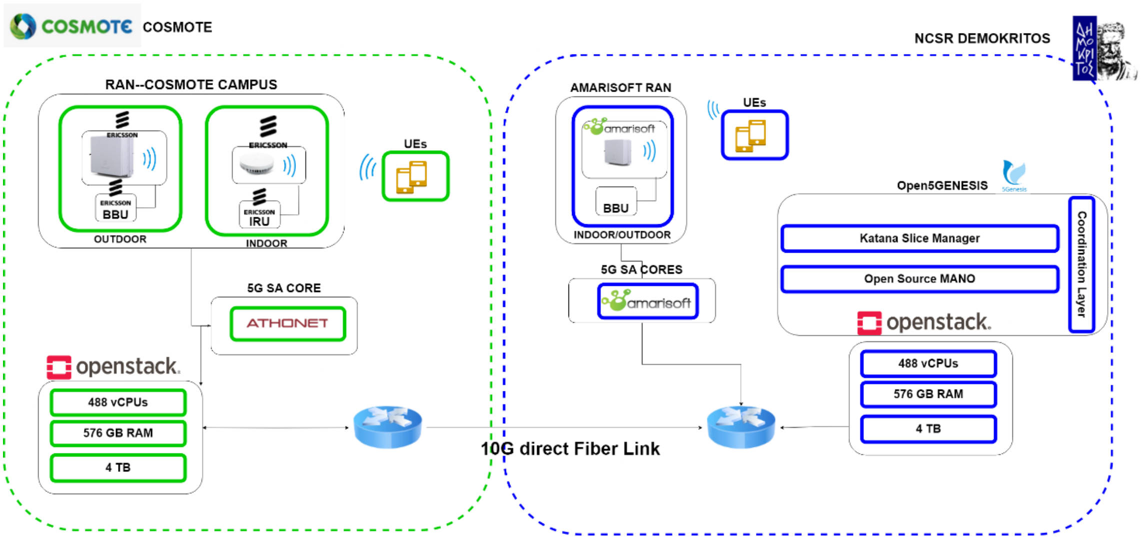

Figure 1. Athens Platform Updated

Although the Open5Genesis framework is well described in the intermediate deliverable [5.1], a brief description is provided below for the reader's convenience (see Fig. 2). The framework is composed of three layers:

- Management and Orchestration (MANO) layer: Handles virtualization, network slices, and virtual resources management.
- Coordination layer: Responsible for the overall coordination of the experiments, including experiments' life cycle management, KPIs monitoring, and analytic results presentation.
- Infrastructure layer: Handles user traffic providing 5G network connectivity.

A set of seven virtual machines have been deployed in a cloud computing infrastructure manager (OpenStack) for the Coordination and MANO layers. The **MANO layer** includes two main components:

- ETSI open-source MANO (OSM): Manages Network Services (NS) and therefore Virtualized network Functions (VNF).
- Open5Genesis Slice Manager: Configured to interconnect with the OSM. Network slice templates (NSTs) are defined in the slice manager, referencing NSs in OSM.

It's important to note that two separate virtual infrastructure managers (OpenStack and Redhat Openstack in NCSRD and COSMOTE respectively) have been integrated with OSM and Slice Manager for the Network Service instantiation upon network slice deployment. These Virtualised Infrastructure Managers (VIMs) are responsible for instantiating the virtual machines required for experiments' execution. The **Coordination layer** consists of five virtual machines, serving each of the Open5Genesis required components for the experiment's coordination:

- Experiment's metrics persistence and graphical presentation component: Uses InfluxDB and Grafana.
- Analytics component: Provides methods for analyzing and offline learning on the data and is responsible for the monitoring of the platform.
- Experiment Life Cycle Manage (ELCM): Schedules and executes experiments.
- Dispatcher component: Entry point of the system, offering the functionalities to an Experimenter through a single interface.

- Open5Genesis Portal: Provides a Graphical User Interface (GUI) for all experiments infrastructure stakeholders in order to create, run, and monitor (in real time) experiments execution.

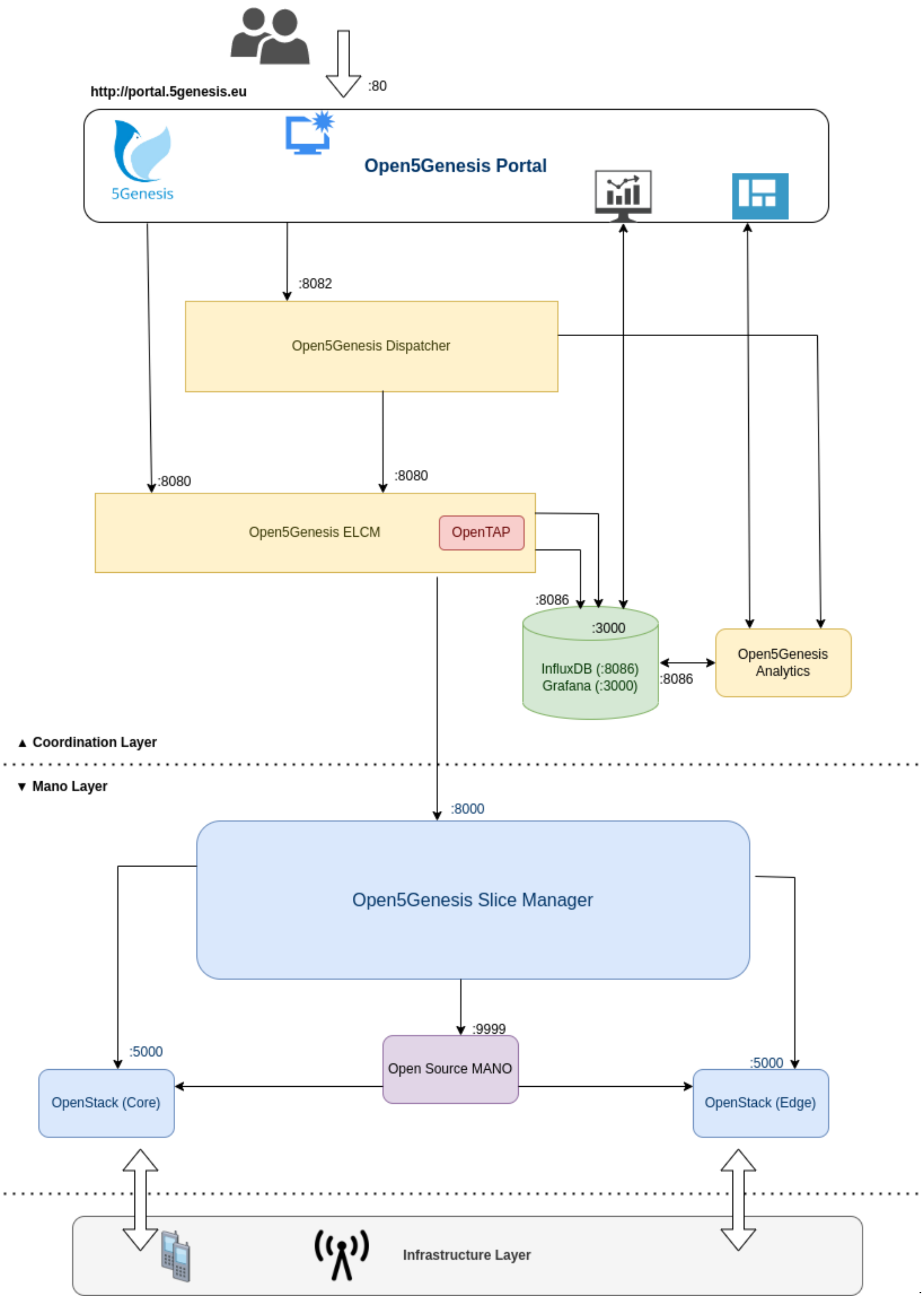

Figure 2. Open5Genesis Experimentation Framework

The 5G systems under test for both sites of the Athens platform, at NCSRD and COSMOTE are described below:

- RAN Systems:
  - The RAN for the COSMOTE testbed is based on the Ericsson small cells solutions (https://www.ericsson.com/en/small-cells/cbrs) and incorporates a baseband

unit (BBU 6630) that provides high-performance connectivity for mobile networks. It is compatible with various radio units, including the 4408, which is designed to provide high-capacity and low-latency connectivity for outdoor deployments. In addition to the radio unit, the system also includes the Indoor Radio Unit (IRU) 8848 and Dot 4479 B78L, which are essential components for the indoor deployment of a 5G network. The GPS system is used for synchronization purposes and ensures accurate timing and location data for network operations. Together, these components form a powerful and reliable radio access network that delivers high-speed connectivity and low latency. The system operates at the 3.5 GHz frequency bands allocated to COSMOTE,

  - The Amarisoft 5G NR, which is supported at the NCSRD site, operates in TDD frequency bands below 6 GHz with up to 50 MHz of bandwidth for the purpose of the experiments. It supports various subcarrier spacing options, FDD bands, and can operate in MIMO configurations up to 2x2 in DL.

- 5G Core: The 5G Core networks, ATHONET for COSMOTE and AMARISOFT for NCSRD, provide essential network functions for the operation of a 5G network, such as Access and Mobility Management Function (AMF), Authentication Server Function (AUSF), Session Management Function (SMF), User plane Function (UPF), UDM (Unified Data Management), and 5G Equipment Identity Register(5G-EIR).

The table below summarizes the technologies used for the final platform assessment and those used in the initial assessment.

Table 1. Technology Comparison for Platform Assessment

Network Domains	Technology Options			
Testbed	**NCSRD**		**COSMOTE**	
Round	**Initial**	**Final**	**Initial**	**Final**
Cloud	Openstack	Openstack	Openstack	RedHat Openstack
MANO	OSM 8	OSM 8	OSM 8	OSM 8
Network Slicing	Katana Slice Manager	Katana Slice Manager	Katana Slice Manager	Katana Slice Manager
3GPP Technology	SA	SA	NSA	SA
5G Core	Amarisoft 5GC	Amarisoft 5GC	Athonet EPC	Athonet 5GC
RAN	Amarisoft gNB	Amarisoft gNB	Nokia Airscale (RRH)	ERICSSON gNB
UE	COTS UEs	COTS UEs	COTS UEs	COTS UEs

A noteworthy update regarding technology from the initially reported assessment is the incorporation of a 5G SA network in COSMOTE instead of the 5G NSA network reported in the initial assessment [D5.1]. For the NCSRD testbed, we assessed a new Amarisoft Classic 5G all-in-one solution (i.e., the system under test remains unchanged). The evaluation was repeated to ensure the stability of the new equipment and to confirm that the results would remain consistent with the initial ones.

2.2.2 The Malaga Platform

The final assessment of the Malaga platform is based on the performance evaluation of the mmWave deployment shown in Figure 3. This deployment includes 2 5G NR TDD cells in FR2 band n258 and 2 4G cells in band 7. The Open5Genesis framework is also used for the management of the experiment execution on top of the infrastructure layer described in Figure 3.

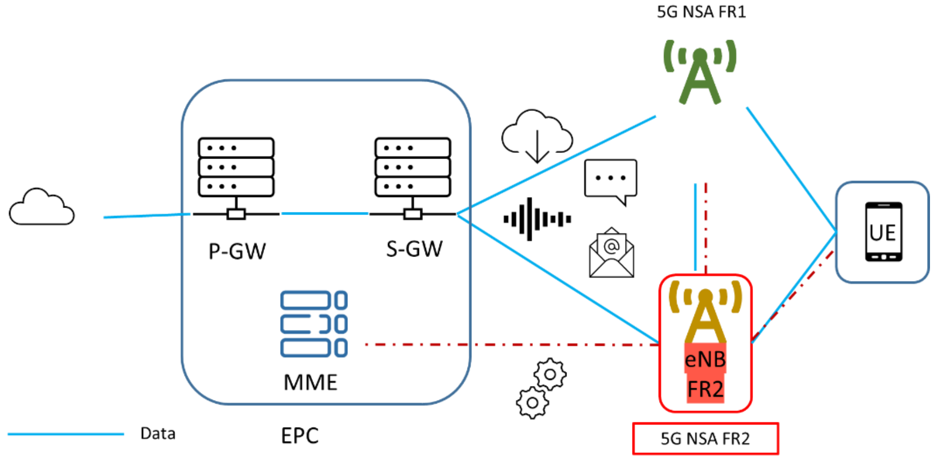

Figure 3. 5G NSA FR2 deployment at Malaga platform

Figure 4. 5G TDD FR2 cell

2.3 Final Tests and Results

2.3.1 Athens Platform Tests and Results

The final assessment of the Athens platform was conducted using the Open5Genesis experimentation framework. The assessment involved both the NCSRD Demokritos and COSMOTE sites including throughput and end-to-end RTT latency tests. Initially, the tests are described in detail using the test case templates shown in Annex 6.1, customized for EVOLVED-5G. Then, the experiments have been executed using the Open5Genesis experimentation framework, and the final results are obtained from the Analytics framework, with high granularity. Figure 5 illustrates the test setup for all experiments between endpoints A and B, as conducted at NCSRD site. The experiments include throughput for both downlink and uplink, and end-to-end RTT for "standard" and low latency RAN configuration (i.e., reducing the period of scheduling request). The COSMOTE site includes throughput and end-to-end RTT latency tests, which are orchestrated by the Open5Genesis experimentation framework, hosted on NCSRD premises, as shown in Figure 6. The interconnection between the two sites is facilitated through a dedicated 10Gbit dark fiber.

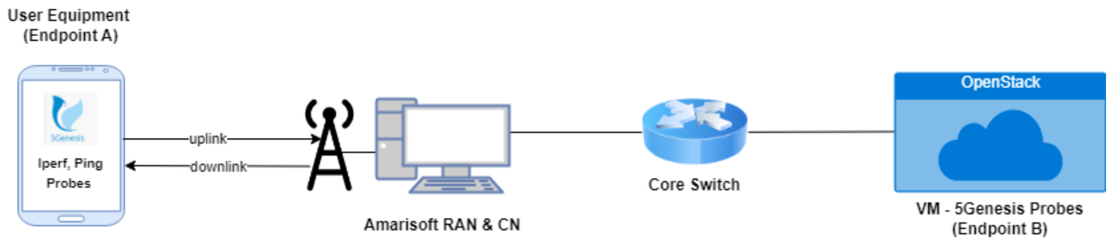

Figure 5. NCSRD site testbed setup

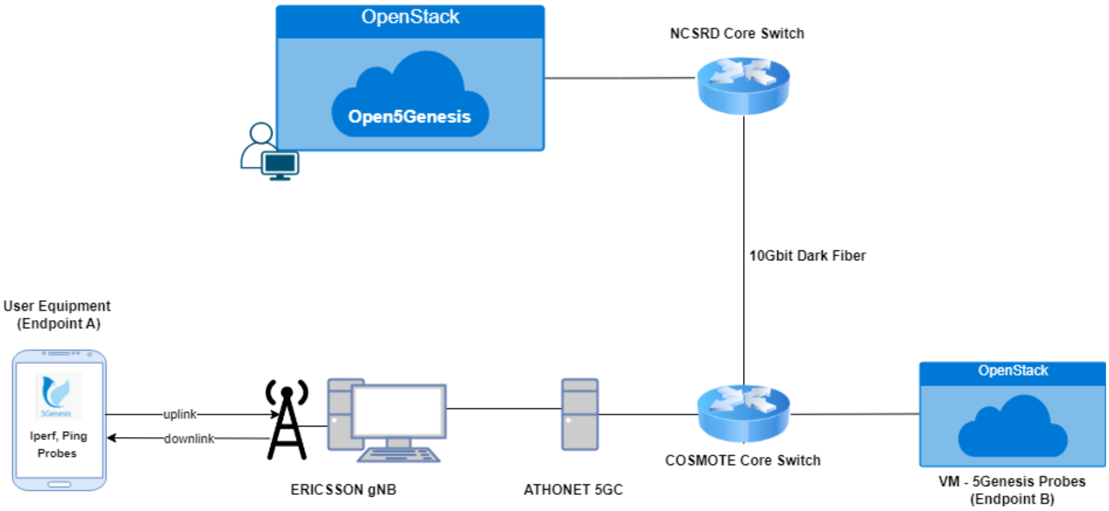

Figure 6. COSMOTE - NCSRD sites testbed setup

2.3.1.1 Throughput

Throughput experiments were run for both the 5G SA deployments supported by the Athens platform and were based on the predefined test templates defined as part of the project's methodology. A more comprehensive description of these experiments can be found in Annex 6.2, which includes information about the scenario, the infrastructure, the targeted KPIs, and the sequence of test cases, outlining the steps for their execution. The radio configurations for NCSRD and COSMOTE are presented in Table 2. More specifically, for:

- Amarisoft SA hosted at NCSRD site, the experiments were carried out using the predefined test case templates NCSRD_Downlink, NCSRD_Uplink, and NCSRD_Best_Uplink, which corresponded to the evaluation of downlink, uplink, and best uplink scenarios, respectively. The System Under Test (SUT) for NCSRD consisted of the commercial Amarisoft Classic, including both 5G-NR and 5GC Rel. 16 components, a COTS UE, and a VM incorporating the 5Genesis iPerf probe. All tests were conducted within a controlled laboratory environment with ideal channel conditions, resulting in an approximate Modulation and Coding Scheme (MCS) value of 26.

- Athonet/Ericsson SA hosted at COSMOTE, the experiments were carried out using the templates COS_Downlink and COS_Uplink for downlink and uplink evaluation respectively. for the SUT for COSMOTE includes Ericsson and Athonet Rel. 16 solutions, a COTS UE and a VM where the probe is deployed. Each experiment was conducted through three iterations, with each iteration comprising 60 samples. During the final assessment, the generated traffic is routed through the Transmission Control Protocol (TCP) , whereas the initial tests employed the User Datagram Protocol (UDP) .

Table 2. Athens Platform Radio Configurations

NCSRD Amarisoft	
Band	n78, ARFCN 632628, 3489.42 MHz
Mode	TDD
Bandwidth	50 MHz
Carrier-Components	1 carrier
MIMO-layer	2T1R
DL MIMO Mode	2X2
Beams	NA
Subcarrier-spacing	30 KHz
Uplink/Downlink slot ratio	7 DL slots, 2 UL slots, 1 special slot
COSMOTE Ericsson	
Band	n78, ARFCN 636666, 3500 MHz
Mode	TDD
Bandwidth	100 MHz
Carrier-Components	1 carrier
MIMO-layer .	4T1R
DL MIMO Mode	4X4
Beams	NA
Subcarrier-spacing	30 KHz
Uplink/Downlink slot ratio	DDDSUUDDDD

Finally, the outcomes of each experiment for NCSRD were compared to a maximum theoretical value derived from equation (1), as defined in 3GPP TS 38.306 [3].

$$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{max} \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \left(1 - OH^{(j)}\right) \right) \quad (1)$$

Where:

- j is the number of the aggregated component carriers which is **1** since one component carrier is used
- $v_{Layers}^{(j)}$ is the maximum number of Multiple-input / multiple output (MIMO) layers, which is **2** for downlink and **1** for uplink
- $Q_m^{(j)}$ is the modulation order, which is **8** considering MCS 26

- $f^{(j)}$ is the scaling factor and can take the values **1**, 0.8, 0.75 and 0.4, which is 1 since there is only one component carrier
- μ is the numerology as defined in TS 38.211, which is **1** for 30 KHz Subcarrier Spacing (SCS)
- T_s^{μ} is the average Orthogonal Frequency Division Multiplexing (OFDM) symbol duration in a subframe for numerology μ, assuming normal cyclic prefix, which is $T_s^{\mu} = \frac{10^{-3}}{14 \cdot 2^{\mu}}$ = 3.571 × 10^{-5} sec ≈ **35 µs**
- $N_{PRB}^{BW(j),\mu}$ is the maximum number of Physical Resource Blocks (PRB) for selected $BW^{(j)}$ with numerology μ, as defined in 5.3 TS 38.101-1 and 5.3 TS 38.101-2
- 12 is the number of subcarriers for 1 PRB
- $OH^{(j)}$ is the overhead for control channels and takes the following values:
  - 0.14, for frequency range FR1 for DL
  - 0.18, for frequency range FR2 for DL
  - 0.08, for frequency range FR1 for UL
  - 0.10, for frequency range FR2 for UL

- R_{max} = 948/1024. However, for MCS 26, R = **916.5**

2.3.1.1.1 DL Throughput results

In the case of downlink throughput, the generic experiment starts with the iPerf client, which operates from the VM hosting the 5Genesis iPerf probe (endpoint B). This client generates TCP traffic directed towards the UE (endpoint A). On the server side, the UMA iPerf Android application functions as an iPerf server, installed on the device responsible for capturing all the essential results. Figures 7 and 8 illustrate the throughput results, measured in Mbps, across three consecutive iterations, each consisting of 60 samples for Amarisoft and Athonet deployments respectively and Tables 2 and 3 present the statistical analysis of the results.
It's noteworthy that the average values for these three iterations are as follows:

- Amarisoft/NCSRD deployment: The results are presented in Table 2. Based on the above equation the theoretical throughput for the corresponding radio configuration described in Annex 6.2 is 386 Mbps consider also that for TDD duplex mode part of the slots allocated for DL is 70%, where 1 = 100% of Slots (3GPP 38.213). Note that in the above calculation process, block error rate probability is not considered, therefore the result of the experiment is sufficiently close to the maximum theoretical value.
- Athonet-Ericsson/COSMOTE deployment: The results are presented in Table 3. The achieved metrics are exceptional when compared to reported measurements in the 5GPP report on trial results [4], that indicate speeds below 700 Mbps.

Table 3. Amarisoft/NCSRD Downlink Throughput Statistical

Mean:	331.3 Mbps
Standard deviation:	47.09 Mbps
Median:	370 Mbps
Max:	372 Mbps
Min:	371 Mbps
Percentile:	Q1 =369.92 Mbps, Q2 = 370 Mbps, Q3 = 371 Mbps

Confidence Interval (5%, 95%):	[347.43, 349.21] Mbps

Table 4. Athonet-Ericsson/COSMOTE Downlink Throughput Statistical

Mean:	905.73 Mbps
Standard deviation:	119.15 Mbps
Median:	916 Mbps
Max:	1048.33 Mbps
Min:	173 Mbps
Percentile:	Q1 = 863.08 Mbps, Q2 = 917.83 Mbps, Q3 = 963.83 Mbps
Confidence Interval (5%, 95%):	[871.335, 875.7619] Mbps

In addition to the initial throughput tests, we have also gathered complementary data pertaining to the transport layer calculated from the Open5Genesis framework. These additional results include both jitter measurements (ms) and packet loss percentage (%) on the server side. It's important to emphasize that when it comes to TCP, there is neither packet loss nor jitter, as TCP inherently manages packet loss, and in our controlled lab environment, the occurrence of packet queuing delays is rare. As mentioned in D5.1 in the initial tests, the high packet loss percentage arises from the fact that UDP bandwidth is set to 400 Mbps (NCSRD testbed), thus the radio channel can handle 338.82 Mbps on average.

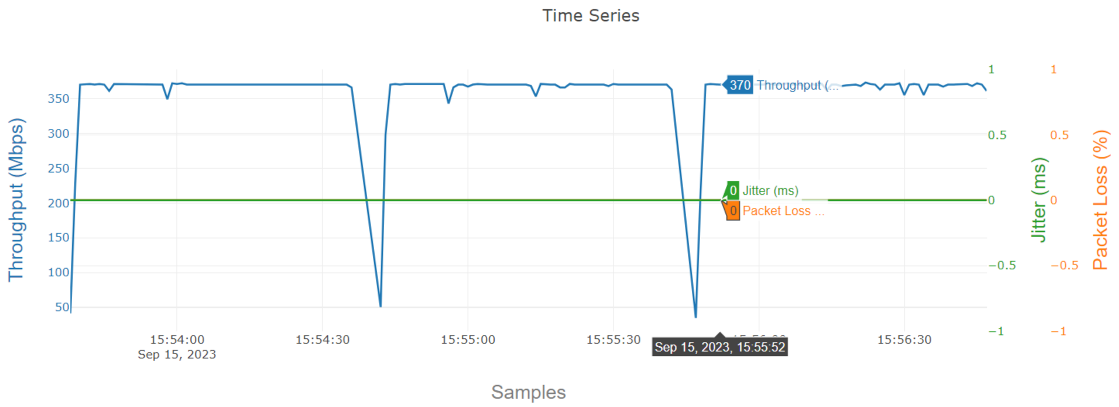

Figure 7. DL throughput results (NCSRD)

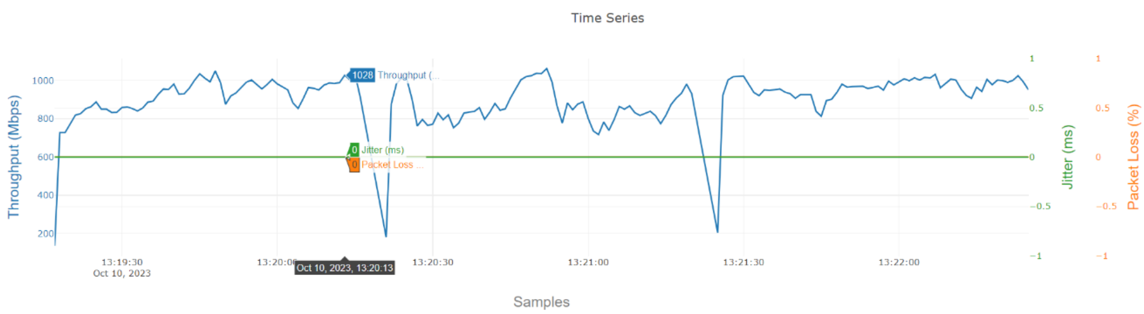

Figure 8. DL throughput results (COSMOTE)

2.3.1.1.2 UL Throughput results

In the uplink direction, the experiments begin with UMA iPerf android application acting as a client, thus the TCP traffic starts from the device (endpoint A) towards the VM (endpoint B).

- For Amarisoft/NCSRD, for uplink throughput evaluation, two distinct experiments have been defined, each corresponding to different slot configurations. The first configuration is the same as in the downlink throughput experiment with 2 uplink slots in a period of 5ms. In order to maximize the uplink speed, the second experiment uses 8 uplink slots in the same 5ms period. The details of the radio configurations are described in Annex 6.2.2 6.2.3. The results of these experiments are depicted in Figure 9 and Figure 10, representing the first and the second (best uplink) configurations, respectively. Additionally, Tables 4 and 5 summarize the statistical analysis of the results. The theoretical values resulting from the above equation are 63.08 Mbps for the first configuration and 232 Mbps for the best uplink, also considering that for TDD duplex mode part of the slots allocated for UL is 20% and 80% respectively. It's worth highlighting that in the best uplink case, throughput is improved by 173.78 Mbps on average (based on the median values). Jitter and packet loss measurements (Annex 6.4) have also been considered in both experiments for uplink, and the justification of the results is the same as in downlink. Note that (for the initial tests), adjusting the UDP bandwidth close to the expected result leads to close to zero packet loss values.
- For Athonet-Ericsson/COSMOTE, in the uplink direction, the experiment is conducted using the fixed TDD frame structure, with two uplink slots and eight downlink slots (DDDSUUDDDD) in a period of 5ms. More details of the radio configurations can be found in Annex 6.2.5. The results of the experiment are presented in Figure 11 and the statistical analysis is provided in Table 5. Considering the TDD frame structure selected, which is bandwidth optimised (since the spectrum is allocated 8:2 downlink to uplink) and the reported 5GPP trial results [4], these values match the performance targets set.

Table 5. Amarisoft/NCSRD Uplink Throughput Statistical

Mean:	48.49 Mbps
Standard deviation:	2.37 Mbps
Median:	48.82 Mbps
Max:	52.36 Mbps
Min:	38.03 Mbps
Percentile:	Q1 = 47.26 Mbps, Q2 = 48.82, Q3 = 49.93 Mbps
Confidence Interval (5%, 95%):	[47.8, 47.89] Mbps

Table 6. Amarisoft/NCSRD best Uplink Throughput Statistical

Mean:	214.18 Mbps
Standard deviation:	15.59 Mbps
Median:	222.6 Mbps
Max:	226 Mbps
Min:	162.3 Mbps
Percentile:	Q1 = 212 Mbps, Q2 = 222.67 Mbps, Q3 = 224 Mbps
Confidence Interval (5%, 95%):	[209.59, 210.19] Mbps

Table 7. Athonet-Ericsson/COSMOTE Uplink Throughput Statistical

Mean:	67.58 Mbps
Standard deviation:	2.96 Mbps
Median:	68.63 Mbps
Max:	70.8 Mbps
Min:	55.73 Mbps
Percentile:	Q1 = 65.85, Q2 = 68.55, Q3 = 69.78 - Mbps
Confidence Interval (5%, 95%):	[66.72, 66.83] Mbps

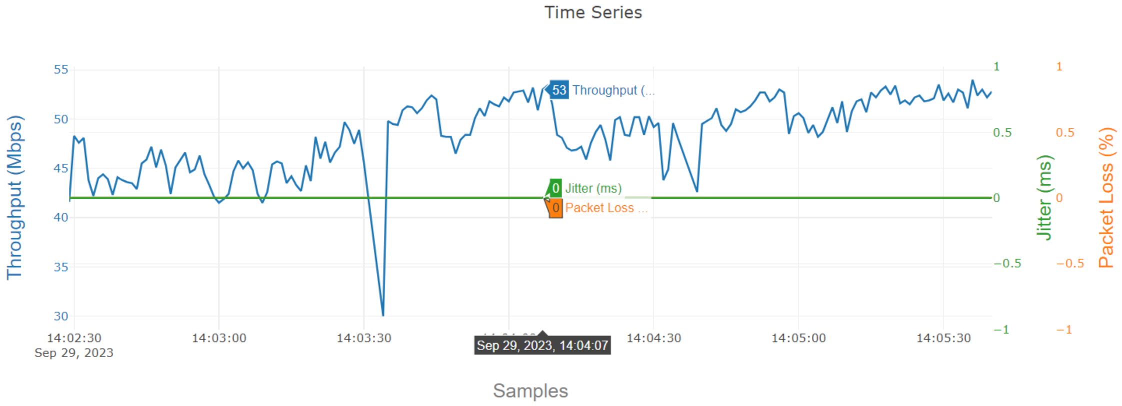

Figure 9. UL throughput results (Amarisoft/NCSRD)

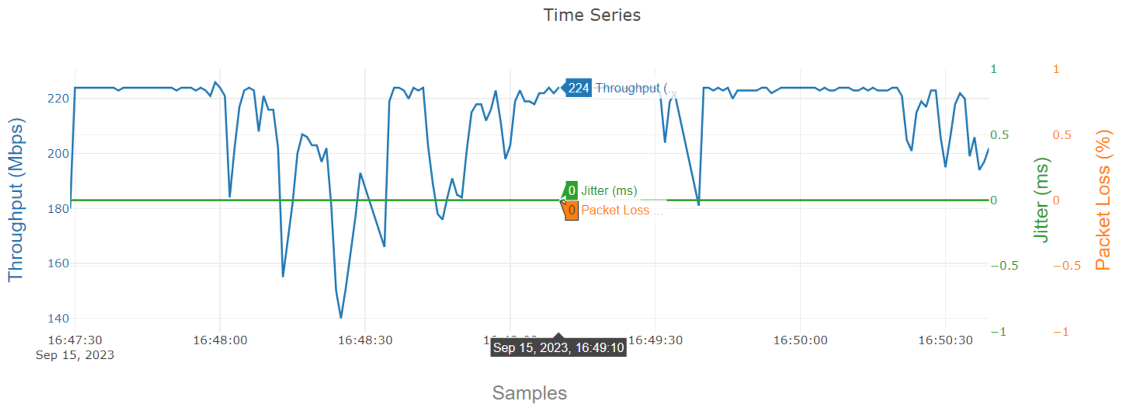

Figure 10. Best UL throughput results (Amarisoft/NCSRD)

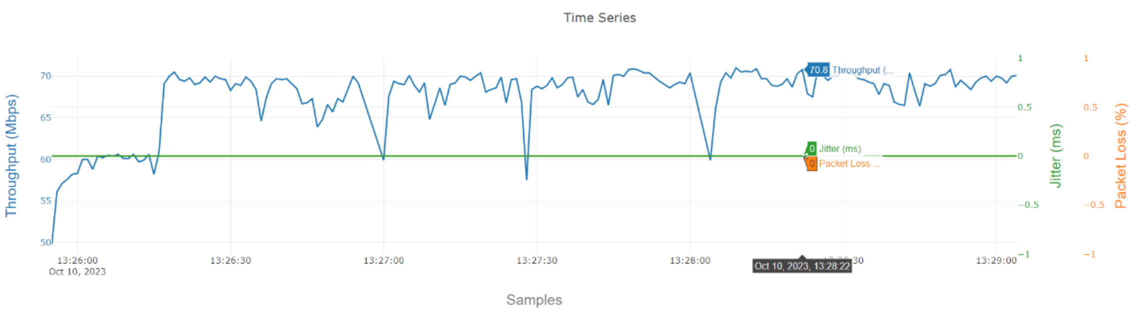

Figure 11. UL throughput results (Athonet-Ericsson/COSMOTE)

2.3.1.2 Round Trip Time results

Delay experiments were executed based on test case templates evaluating the RTT between a UE and the VM deployed on OpenStack. More details are described on Annex 6.2 including the scenario, the testing infrastructure, the target KPI and the test case sequence along with the execution steps.

- For Amarisoft/NCSRD, as in throughput experiments, the SUT involves the commercial Amarisoft Classic (i.e., both 5G-NR and 5GC Rel. 16), one COTS UE and a VM that integrates the 5Genesis ping probe. All the tests were conducted in a lab environment with perfect channel conditions leading to an approximate 26 MCS value and a packet size of 64 byte. Two experiments are defined using different scheduling request periods. For the low latency experiment the scheduling request period has been reduced from 10 to 0.5 ms and symmetric slot configuration is used. Figures 12 and 13 present the results for 64byte packet size and Table 7 provide the statistical values. It is worth noting that for the low latency configuration the mean value is reduced by 18.7 ms.
- For Athonet-Ericsson/COSMOTE, the initial RTT experiment was conducted considering the multi domain deployment, including both COSMOTE and Demokritos testbeds that are interconnected. Specifically, the UE was operating in COSMOTE, and the VM with the 5Genesis ping probe was deployed on NCSRD (i.e., OpenStack). Therefore, the traffic was generated from NCSRD's ping probe, traversing the network through the GRNET, reaching COSMOTE's UE, and then returning to the ping probe. During the final tests, the experiment follows the topology depicted in Figure 6, where both the UE and the VM with the ping probe are located in COSMOTE's testbed. As mentioned, the experiment is managed through the Open5GENESIS framework hosted in NCSRD. The final results are presented in Figure 14 and Table 8.

Table 8. Amarisoft/NCSRD RTT Statistical

RTT (ms)	
Mean:	28.69 ms
Standard deviation:	35.32 ms
Median:	26.11 ms
Max:	52.06 ms
Min:	12.63 ms
Percentile:	Q1 = 19.48 ms, Q2 = 24.45, Q3 = 29.26 ms
Confidence Interval (5%, 95%):	[18.2, 19.57] ms
Low - RTT (ms)	
Mean:	9.99 ms
Standard deviation:	17.89 ms
Median:	7.53 ms
Max:	144.16 ms
Min:	6.2 ms
Percentile:	Q1 = 6.99 ms, Q2 = 7.54 ms, Q3 = 8.14 ms
Confidence Interval (5%, 95%):	[4.68, 5.37] ms

Table 9. Athonet-Ericsson/COSMOTE RTT Statistical

Mean:	15.78 ms
Standard deviation:	5.24 ms
Median:	15.23 ms
Max:	46.16 ms
Min:	7.16 ms
Percentile:	Q1 = 12.61 ms, Q2 = 15.23 ms, Q3 = 18.18 ms
Confidence Interval (5%, 95%):	[14.48, 14.65] ms

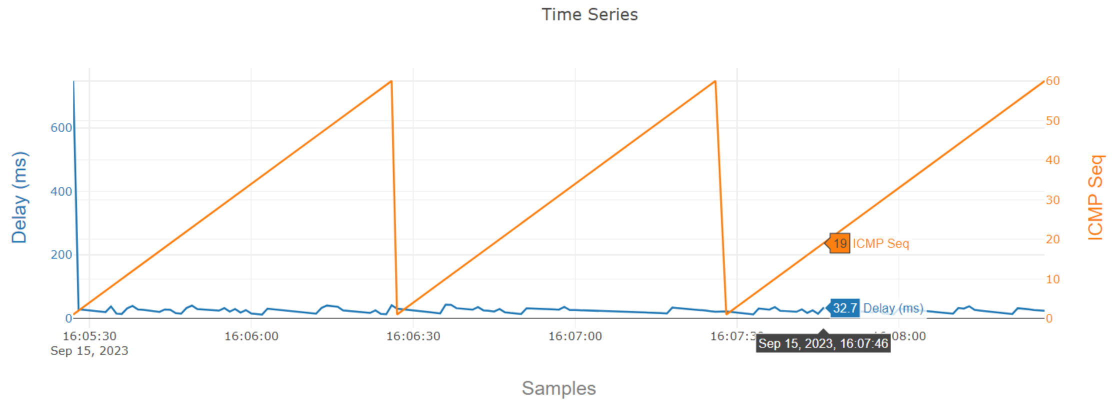

Figure 12. RTT - 64byte packet size (Amarisoft/NCSRD)

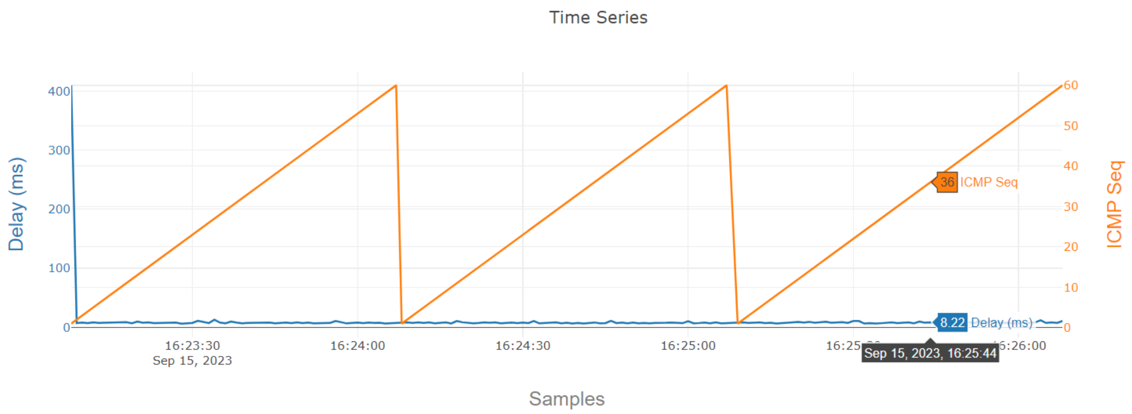

Figure 13. RTT – low latency - 64byte packet size (Amarisoft/NCSRD)

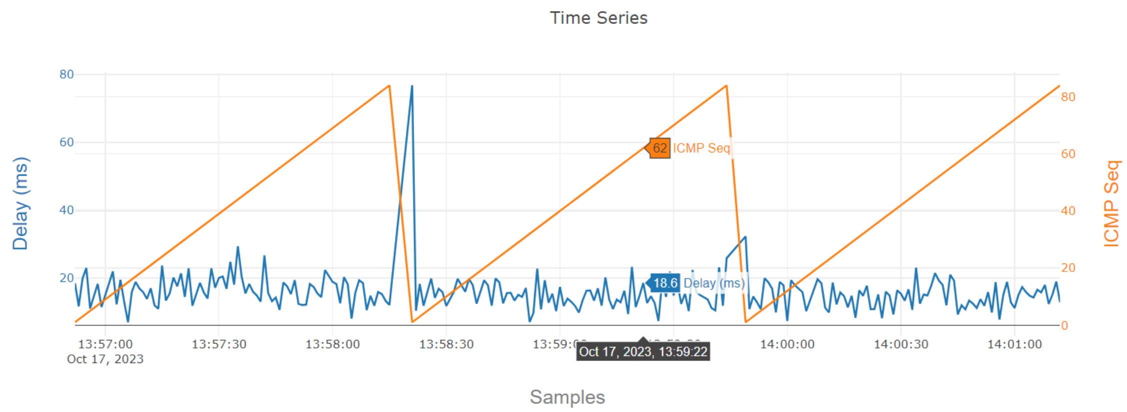

Figure 14. RTT - 64byte packet size (Athonet-Ericsson/COSMOTE)

2.3.2 Málaga Platform Tests and Results

The experiments performed on the UMA platform for performance testing consist of RTT latency and downlink throughput measurements for the 5G NSA outdoor FR2 mmWave and Delay and Jitter measurements for the TSN standard set. Table 9 summarizes the configuration applied in the network during the assessment. Each one of the cells has an associated channel bandwidth of 100 MHz and a MIMO-layer configuration of 2T2R and 32 beams.

Table 10. 5G NSA FR2 Configuration at UMA testbed

Band	n257
Mode	TDD
Bandwidth	100MHz
Carrier-Components	8 carriers
MIMO-layer	2T2R
DL MIMO Mode	2x2 Closed Loop Spatial Multiplexing
Beams	32 beams
Subcarrier-spacing	240KHz
Uplink/Downlink slot ratio	1/4

2.3.2.1 DL Throughput Results

Throughput DL performance tests have been run between the main compute node of the testbed and a 5G UE based on the UMA iPerf agents. The iPerf server is deployed in the UE and the iPerf client is running on the main compute node. The test execution has been automated using Open5GENESIS Suite and OpenTAP. The UE used with mmWave FR2 compatibility is arranged on a fixed outdoor stand.

Figure 16 shows the distributions, using boxplot, of the samples taken in 20 iterations of the iPerf UDP process with the maximum throughput obtained in the network of approximately 1.3Gbps at Packet Data Convergence Protocol (PDCP) level. As can be seen, the measurements taken from DL throughput UDP reflect a maximum feasible for FR2 mmWave network characterization with a payload traffic volume of between 1.3 Gbps and 1.4 Gbps.

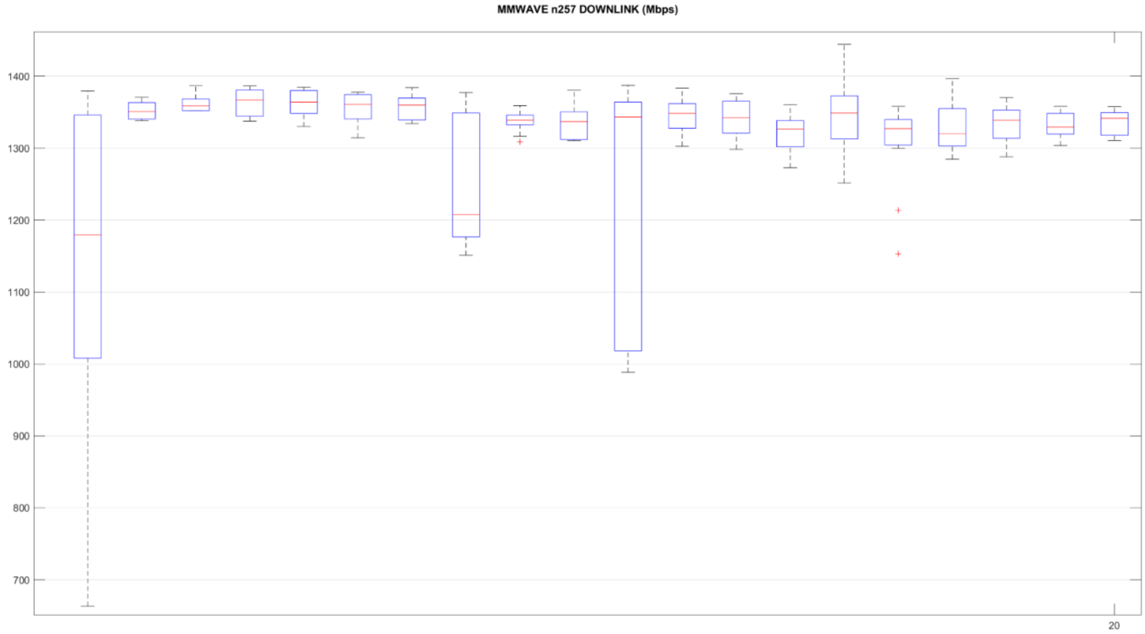

Figure 15. 5G NSA FR2 throughput per iteration (UMA)

Statistical analysis of the results of the 20 iterations of the UDP throughput measurement process provides a T Student confidence interval of between 1292.8227 Mbps to 1328.4082 Mbps with a maximum obtained of 1444.2 Mbps. It is provided in Table 10.

Table 11. 5G NSA FR2 throughput statistical analysis (UMA)

Mean:	1310.6154 Mbps
Standard deviation:	139.9251 Mbps
Median:	1344.15 Mbps
Max:	1444.2 Mbps
Min:	89.6 Mbps
Percentile:	Q1 = 1316.2, Q2 = 1344.15, Q3 = 1360.05 Mbps
Confidence Interval (5%, 95%):	[1292.8227, 1328.4082] Mbps

The signal quality in the DL flow measurement process is characterized by the KPIs of reference Reference Signal Received Power (RSRP) signal power value and received noise in dBm and Received Singal Received Quality (RSRQ) as reference signal quality value in dB. The statistics of these measurements are shown in Table 11.

Table 12 . Complementary radio measurements for 5G NSA FR2 throughput tests (UMA)

NR RSRP max	-70,5 dBm
NR RSRP min	-101.2 dBm
NR RSRP avg	-83.7 dBm
NR RSRQ max	-10.3 dB
NR RSRQ min	-11.5 dB
NR RSRQ avg	-10.7 dB

2.3.2.2 Round Trip Time results

The tests are based on the UMA Ping agent and have been performed between the main computing node and the 5G UE. The tests have been automated using Open5GENESIS Suite. The configured Internet Control Message Protocol (ICMP) packet size is 56 bytes. A total of 20 iterations with a duration of 180 seconds have been executed. Figure 16 shows the sample distribution of the 20 RTT measurement iterations by boxplot.

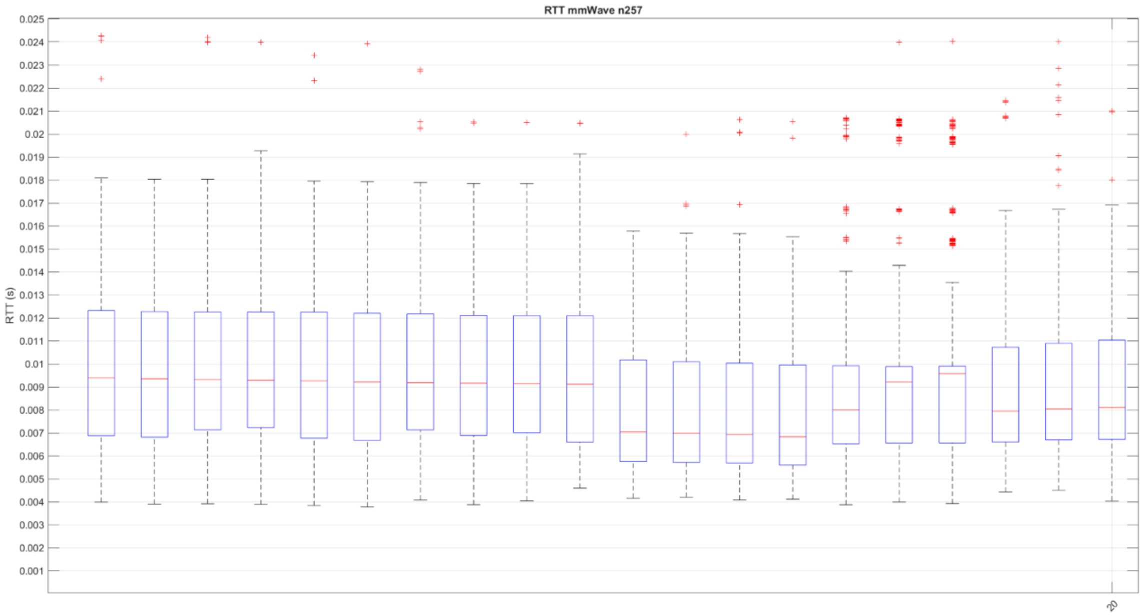

Figure 16. Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT per iteration (UMA)

Statistical analysis of the RTT measurement results show a Student-T confidence interval of 9.3838 ms - 9.4726 ms and a minimum RTT of 3.777 ms shown in Table 9. The statistical analysis of the results is provided in Table 9.

Table 13. 5G NSA FR2 RTT statistical analysis (UMA)

Mean:	0.0094282 s
Standard deviation:	0.0042971 s
Median:	0.009251 s
Max:	0.12979 s
Min:	0.003777 s
Percentile:	Q1 = 0.006707, Q2 = 0.009251, Q3 = 0.011452 s
Confidence Interval (5%, 95%):	[0.0093838, 0.0094726] s

The analysis of the measurements also shows a stable behavior of the infrastructure regarding the delay.
The signal quality in the RRT measurement process is characterized by the KPIs of reference RSRP signal power value and received noise in dBm and RSRQ as reference signal quality value in dB. The statistics of these measurements are shown in Table 10.

Table 14. Complementary radio measurements for 5G NSA FR2 RTT tests (UMA)

NR RSRP max	-70,5 dBm
NR RSRP min	-101.2 dBm
NR RSRP avg	-83.7 dBm
NR RSRQ max	-10.3 dB
NR RSRQ min	-11.5 dB
NR RSRQ avg	-10.7 dB

2.3.2.3 TSN over 5G results

Delay and jitter experiments were carried out using predefined test case templates: UMA_TSN_OWD and UMA_TSN_Jitter, which corresponds to the evaluation of the Downlink One-Way Delay (OWD) and jitter on the Time Sensitive Networking (TSN) over 5G architecture, respectively. A more comprehensive description of these experiments can be found in Annex 6.3, which includes information about the scenario, the infrastructure, the targeted KPIs, and the sequence of test cases, outlining the steps for their execution. All tests were performed in the TSN over 5G setup depicted in Figure 17 (see [24] for further details), which includes the TSN endpoints, the TSN translators (NW-TT and DS-TT), a 5G UE and the 5G network (RAN + 5GC).

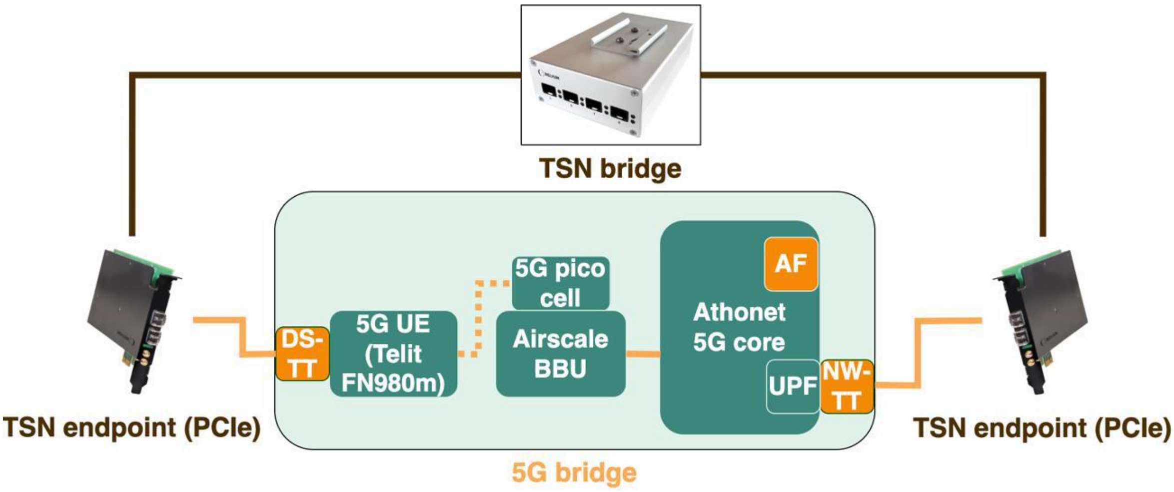

Figure 17. UMA - TSN testbed setup

It should be noted that for each test case a table with results and a figure are provided. Each experiment was carried out by 24 iterations of 1 hour duration, and each iteration is composed of 360 samples (1 sample every 10s). In addition, for comparison purposes, the same experiments have been performed again without using the TSN translators. With these results, the performance of TSN translators and the impact they have on traffic can be tested.

2.3.2.3.1 One-Way Delay

Figure 18 and Figure 19 show the results of the 24 iterations corresponding to the DL OWD (ns) in the TSN configuration over 5G using the TSN translators and without using the TSN translators, respectively.

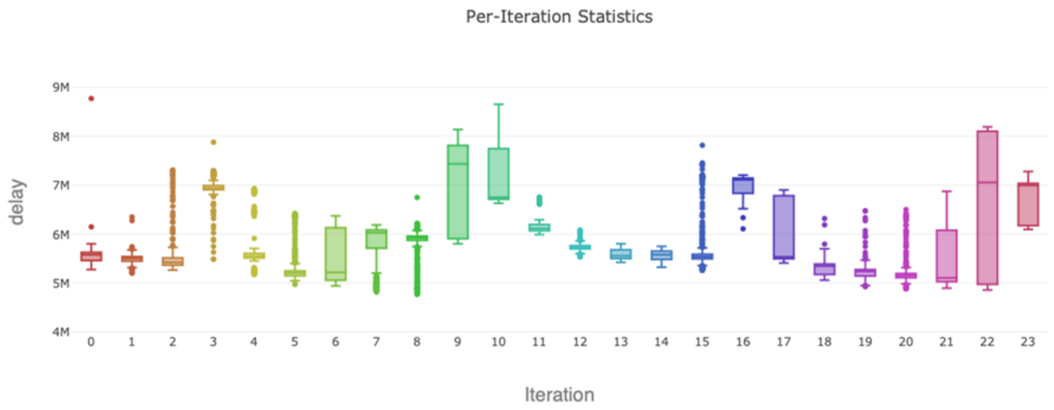

Figure 18. One-Way Delay per iteration in TSN scenario

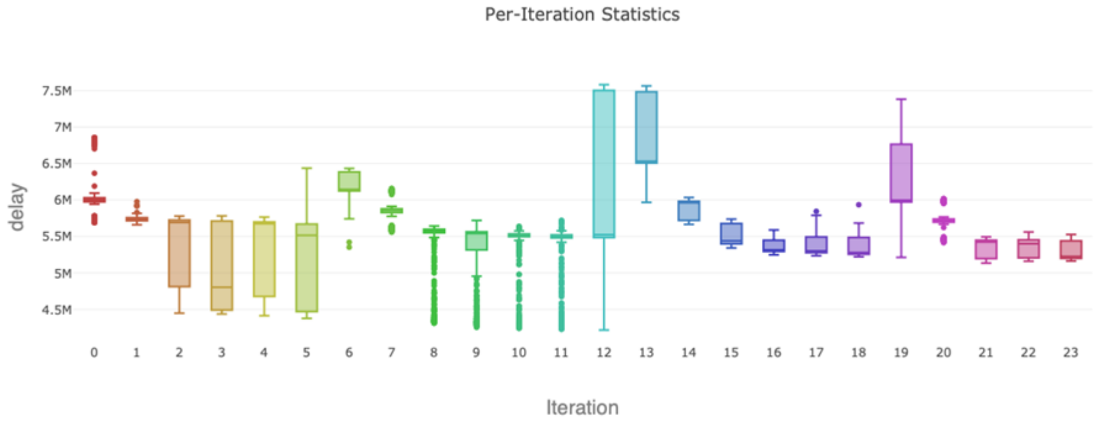

Figure 19. One-Way Delay (ns) per iteration in TSN scenario (without TSN translators)

In addition, Table 15 and Table 16 summarize the statistical analysis of both scenarios. Compared to the scenario in which the TSN translators are not used, the delay is slightly higher, which is to be expected since the TSN translators are currently running on a P4 software switch. However, the delay introduced is only 0.3ms for both TSN translators and is expected to be reduced to values close to 0 once the TSN translators are running on a P4 hardware switch.

Table 15. One-Way Delay (ms) - TSN scenario

Mean:	5.859 ms
Standard deviation:	0.69115 ms
Median:	5.675 ms
Max:	8.1279 ms
Min:	0 ms
Percentile:	Q1 = 5.3578, Q2 = 5.675, Q3 = 6.163 ms
Confidence Interval (5%, 95%):	[5.8444, 5.8735] ms

Table 16. One-Way Delay (ms) - TSN scenario (without TSN translators)

Mean:	5.4396 ms
Standard deviation:	0.68956 ms

Median:	5.3993 ms
Max:	9.9629 ms
Min:	4.3438 ms
Percentile:	Q1 = 4.9356, Q2 = 5.3993, Q3 = 5.8316 ms
Confidence Interval (5%, 95%):	[5.4251, 5.4541] ms

2.3.2.3.2 Jitter

Figure 20 and Figure 21 show the results of the 24 iterations corresponding to the Jitter (ms) in the TSN configuration over 5G using the TSN translators and without using the TSN translators, respectively.

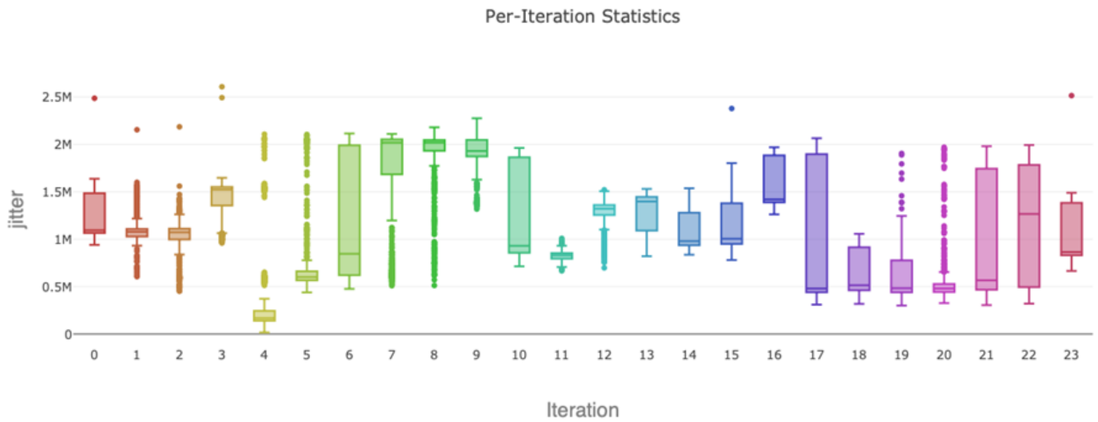

Figure 20. Jitter (ns) per iteration in TSN scenario

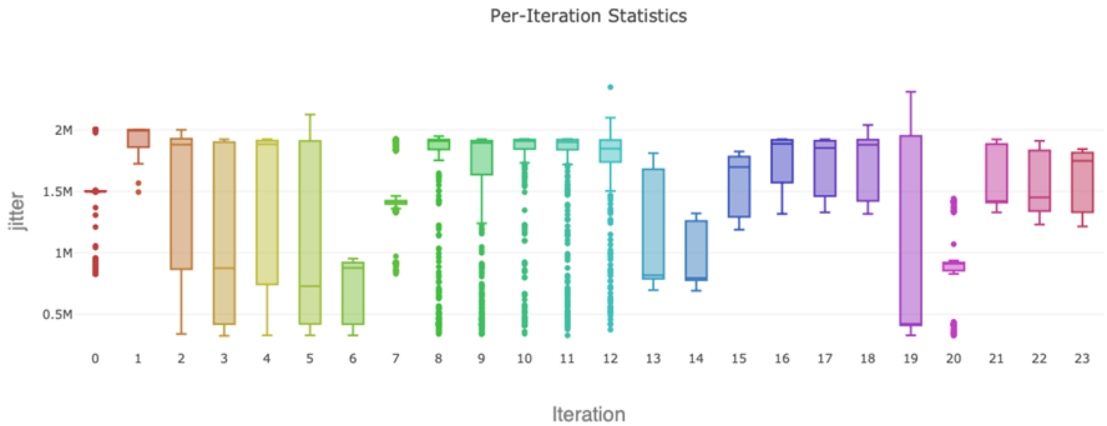

Figure 21. Jitter (ns) per iteration in TSN scenario (without TSN translators)

In addition, Table 17 and Table *18*. Jitter (ms) - TSN scenario (without TSN translators)Table 18 summarize the statistical analysis of both scenarios. In this case, the results obtained using the TSN translators are better than without using them. This is due to the benefits introduced by the TSN translators, for example, the time synchronization allows the variability between delays to be more stable, i.e. to obtain a lower jitter.

Table 17. Jitter (ms) - TSN scenario

Mean:	1.0628 ms
Standard deviation:	0.50558 ms
Median:	1.035 ms
Max:	2.082 ms
Min:	0 ms
Percentile:	Q1 = 0.61675, Q2 = 1.035, Q3 = 1.4911 ms
Confidence Interval (5%, 95%):	[1.0522, 1.0735] ms

Table 18. Jitter (ms) - TSN scenario (without TSN translators)

Mean:	1.0882 ms
Standard deviation:	0.58944 ms
Median:	1.21 ms
Max:	2.0225 ms
Min:	0.00060986 ms
Percentile:	Q1 = 0.59282, Q2 = 1.21, Q3 = 1.6071 ms
Confidence Interval (5%, 95%):	[1.0758, 1.1006] ms

3 COMPONENT-LEVEL EVALUATION

3.1 EVOLVED-5G SOFTWARE COMPONENTS

NEF and CAPIF are two key software components developed in the project to support service exposure and accessibility. The details of the architectural concerns, implementation aspects and the technologies used to develop the NEF emulator can be found on D3.1 [9]. In addition, a thorough description of NEF APIs, including the two APIs that are currently supported can be found on D4.1 [11]. The NEF emulator exposes Northbound APIs to NetApps following the 3GPP TS 29.522 [4] specifications. CAPIF Core Function has been developed following 3GPP TS 23.222 [5] and TS 29.222 [6]. This module is described in deliverables D4.2 [8] and D3.2 [10]. CAPIF Core Function tool has been incorporated to Athens and Málaga platforms as the API Exposure Layer to expose platform APIs, namely, NEF Emulator APIs.

CAPIF offers API management services for API Invokers, in our case, NetApps. The main services offered are:

- API Invoker registration services, which allows NetApps to register in CAPIF Core Function to consume CAPIF services.
- API Publish service, which allows API Exposure services, in our case, NEF Emulator, to publish their APIs for NetApps to discover them.
- API Discovery Service, that allows NetApps to Discover APIs registered in CAPIF, such as NEF Emulator APIs.

CAPIF Core Function has been developed following 3GPP CAPIF APIs YAMLs published in a GitHub repository [13] where all Release 17 3GPP APIs are published. In order to guarantee that API contracts and CAPIF Core Function functionality works properly, a number of tests cases have been developed and automated, so that in every deployment of CAPIF Core Function during Validation or Certification processes, behavior and compliance with standards of CAPIF Core Function is guaranteed.

In order to ensure the proper functionality of these APIs, various test cases have been defined, developed and executed, as described in Section 3.2 (Functional Testing). On top of that, performance tests are also considered including the access time and the success rate of multiple requests. The details of the definition and the results of these sets of tests are described in section 3.3 (Performance Testing)

3.2 FUNCTIONAL TESTING

3.2.1 NEF Emulator results

The NEF emulator constitutes the backend entity that exposes the necessary NEF APIs for facilitating the development of a Network Application without the requirement of NEF exposure availability through a real 5G network. For the NEF emulator that has been developed in the scope of the project, a testing plan has also been introduced (https://github.com/EVOLVED-5G/NEF_emulator/tree/main/docs/test_plan).

The testing plan targets the *MonitoringEvent* and *AsSessionWithQoS* APIs, which are exposed by NEF and exploited by the EVOLVED-5G NetApps. The list of tests defined is presented in the following table:

Table 19. Testing plan targeting the MonitoringEvent and AsSessionWithQoS APIs

TEST	Entity	NEF API
Create subscription by Authorized NetApp	NEF_API_MONITORING_EVENT _API	201 NetApp creates a subscription successfully to the Monitoring Event API for a registered UE.
One-time request to the Monitoring Event API by Authorized NetApp	NEF_API_MONITORING_EVENT _API	200 NetApp sends a one-time response request to the Monitoring Event API for a registered UE.
Create subscription when there is already an active subscription for a registered UE	NEF_API_MONITORING_EVENT _API	409 Conflict / There is already an active subscription for UE with external id 'externalId'.
Create subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT _API	401 Unauthorized.
Read all active subscriptions by Authorized NetApp	NEF_API_MONITORING_EVENT _API	200 With a list of active subscriptions from the Monitoring Event API.
Read all active subscriptions by Authorized NetApp (no active subscriptions available)	NEF_API_MONITORING_EVENT _API	204 No Content.
Read individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT _API	200 Individual subscription by the NetApp from Monitoring Event API is successfully retrieved.
Read individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT _API	404 Not Found.
Read all active subscriptions by unAuthorized NetApp	NEF_API_MONITORING_EVENT _API	401 Unauthorized
Read individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT _API	401 Unauthorized
Update individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT _API	200 Individual subscription by the NetApp from Monitoring Event API is successfully updated.
Update individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT _API	404 Not Found

Update individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT _API	401 Unauthorized
Delete individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT _API	200 Individual subscription by the NetApp from Monitoring Event API is successfully deleted.
Delete individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT _API	404 Not Found
Delete individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT _API	401 Unauthorized
Create subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	201 The NetApp created a subscription successfully to the AsSessionWithQoS for a registered UE.
Create subscription when there is already an active subscription for a registered UE	NEF_API_AS_SESSION_WITH_QOS _API	409 Conflict / There is already an active subscription for UE with (ipv4, ipv6, mac address)
Create subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	401 Unauthorized
Read all active subscriptions by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	200 with subscriptions retrieved successfully by the NetApp from the AsSessionWithQoS API
Read all active subscriptions by Authorized NetApp (no active subscriptions available)	NEF_API_AS_SESSION_WITH_QOS _API	404 Not Found
Read individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully retrieved
Read individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS _API	404 Not Found
Read all active subscriptions by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	401 Unauthorized
Read individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	401 Unauthorized

Update individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully updated
Update individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS _API	404 Not Found
Update individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	401 Unauthorized
Delete individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully deleted
Delete individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS _API	404 Not Found
Delete individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS _API	401 Unauthorized

The functional tests of NEF have been implemented taking advantage of the Robot Framework [12]. They are available in the GitHub repository of NEF-Validation under the branch “capif” (https://github.com/EVOLVED-5G/NEF-Validation/tree/capif). Additional tests have also been implemented to test the rest of the exposed endpoints available by the NEF emulator as listed below:

- Login API
- Cells API
- Default API
- Movement API
- Paths API
- Qos Information API
- UE API
- UI API
- Users API
- gNB API

In this branch of the repository, three folders exist, each one related to some aspect of NEF testing:

- **Pipelines/**: This folder contains the Jenkins pipeline for automatically deploying NEF services if necessary (i.e., if it is not already deployed) and for running the Robot Framework tests.
- **Tests/**: This folder contains the actual tests, including the code of the tests, the test cases, the relevant resources and custom Python libraries. In particular:
    - **Libraries/**: auxiliary code for testing to cover Robot Framework functionalities.
    - **Resources/**: configuration parameters, mainly keywords and variables referenced by the tests.
- **Tools/**: This folder contains the code for containerizing the tests, i.e., code and configuration parameters to generate the Robot Docker image (to be used by Jenkins pipelines) and for

deploying a Jenkins pipeline that uploads the Robot Docker image to the JFrog Artifactory of EVOLVED-5G. Additionally, in this folder, all the necessary files for the registration process to the CAPIF instance are hosted.

The test cases are categorized by the specific API of NEF, i.e., tests/<API_NAME>. Inside that directory a file named /<API_NAME>.robot contains the code for testing each endpoint of the respective API.

The generation of a robot image is realized through the execution of the following command in the path /tools/robot:

```
docker build –no-cache -t ${ROBOT_IMAGE_NAME}:${ROBOT_VERSION} .
```

This command builds a Docker image with the name and version provided. An example follows:

```
docker build –no-cache -t dockerhub.hi.inet/dummy-netapp-testing/robot-test-image:2.0 .
```

If Jenkins is desired to be used, the image is built and then it gets pushed to dockerhub.hi.inet through the pipeline defined in the directory /tools/robot too, which is named robot-image.groovy.

Once the image is built, Robot Framework tests can be executed either locally or remotely via Jenkins. The first option is used during the development to test NEF faster by deploying the Docker image at the system where the development takes place. The second option takes advantage of the EVOLVED-5G CI/CD platform and can be configured to use a deployment of NEF wherever it is available. The deployment of the Docker image is similar with a differentiation in launching the respective container in the second case, as illustrated below:

The Docker command consists of the following:

- "docker run -t –network="host" –rm -d": this will invoke Docker and set some useful options:
  - Run: launch Docker image on Docker.
  - -t: use tty
  - -name: gives the name to the container created.
  - –network: this means Docker image will use the same networks than host.
  - –rm: this option will remove the image after the end of the execution of tests from the Docker environment. Reduces resource usage.
  - -d: this option will make the container to start in detached mode

- "-v ${ROBOT_<DIR>_DIRECTORY}:/opt/robot-tests/<DIR>": these options will attach local directories to volumes defined in the Robot Framework Docker image. In particular:
  - /opt/robot-tests/tests: At this volume Docker image will expect to attach tests directory of repository, including all robot code.
  - /opt/robot-tests/results: At this volume, Robot Framework will store reports generated after the execution of all the tests.
  - /opt/robot-tests/library: At this volume, the Docker image is expected to attach the library directory of the repository.
  - /opt/robot-tests/resources: At this volume, the Docker image is expected to attach the resources directory of the repository.

- /opt/robot-tests/capif-registration: At this volume, the Docker image is expected to attach all the files as they were created during the registration process to the CAPIF API.

- "—env <ROBOT_ENV_VARIABLE>:<ENV_VARIABLE_VALUE>": these options will set environment variables necessary for the robot container.
  - NEF_URL: The host of the NEF emulator instance.
  - BUILD_NUMBER: Jenkins distinct build number to distinguish resources used/created during each test run.
  - NGINX_HOSTNAME: Same as NEF_URL
  - ADMIN_USER: NEF Emulator's admin username.
  - ADMIN_PASS: NEF Emulator's admin password.
  - CERTS_PATH: Path where CAPIF registration certificates are located.
  - CAPIF_HOST: CAPIF API URL.
  - CAPIF_HTTPS_PORT: the port of the CAPIF API exposed for secure connection (HTTPS).

- "${ROBOT_IMAGE_NAME}:${ROBOT_VERSION}": This part indicates the Docker image and version previously defined in the Docker image generation. In the example, the Docker is pulled from the EVOLVED-5G artifactory.

- Options after Robot Image selection:
  - -c: This allows the execution of commands inside the container created.

After running all the tests, the following report is produced:

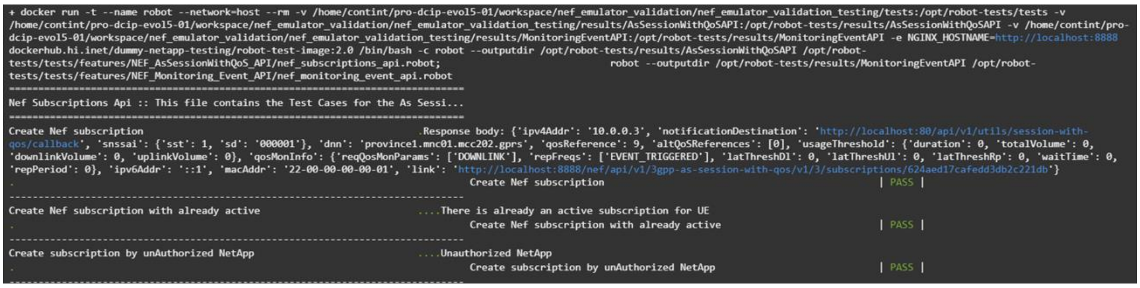
```
+ docker run -t --name robot --network=host --rm -v /home/contint/pro-dcip-evol5-01/workspace/nef_emulator_validation/nef_emulator_validation_testing/tests:/opt/robot-tests/tests -v
/home/contint/pro-dcip-evol5-01/workspace/nef_emulator_validation/nef_emulator_validation_testing/results/AsSessionWithQoSAPI:/opt/robot-tests/results/AsSessionWithQoSAPI -v /home/contint/pro-
dcip-evol5-01/workspace/nef_emulator_validation/nef_emulator_validation_testing/results/MonitoringEventAPI:/opt/robot-tests/results/MonitoringEventAPI -e NGINX_HOSTNAME=http://localhost:8888
dockerhub.hi.inet/dummy-netapp-testing/robot-test-image:2.0 /bin/bash -c robot --outputdir /opt/robot-tests/results/AsSessionWithQoSAPI /opt/robot-
tests/tests/features/NEF_AsSessionWithQoS_API/nef_subscriptions_api.robot;                    robot --outputdir /opt/robot-tests/results/MonitoringEventAPI /opt/robot-
tests/tests/features/NEF_Monitoring_Event_API/nef_monitoring_event_api.robot
==============================================================================
Nef Subscriptions Api :: This file contains the Test Cases for the As Sessi...
==============================================================================
Create Nef subscription                                     .Response body: {'ipv4Addr': '10.0.0.3', 'notificationDestination': 'http://localhost:80/api/v1/utils/session-with-
qos/callback', 'snssai': {'sst': 1, 'sd': '000001'}, 'dnn': 'province1.mnc01.mcc202.gprs', 'qosReference': 9, 'altQoSReferences': [0], 'usageThreshold': {'duration': 0, 'totalVolume': 0,
'downlinkVolume': 0, 'uplinkVolume': 0}, 'qosMonInfo': {'reqQosMonParams': ['DOWNLINK'], 'repFreqs': ['EVENT_TRIGGERED'], 'latThreshDl': 0, 'latThreshUl': 0, 'latThreshRp': 0, 'waitTime': 0,
'repPeriod': 0}, 'ipv6Addr': '::1', 'macAddr': '22-00-00-00-00-01', 'link': 'http://localhost:8888/nef/api/v1/3gpp-as-session-with-qos/v1/3/subscriptions/624aed17cafedd3db2c221db'}
.                                                               Create Nef subscription                                | PASS |
------------------------------------------------------------------------------
Create Nef subscription with already active                 ....There is already an active subscription for UE
.                                                               Create Nef subscription with already active            | PASS |
------------------------------------------------------------------------------
Create subscription by unAuthorized NetApp                  ....Unauthorized NetApp
.                                                               Create subscription by unAuthorized NetApp             | PASS |
------------------------------------------------------------------------------
```

Figure 22. NEF Emulator Results – Test Report

3.2.2 CAPIF Tool results

As described in the introduction, along with CAPIF Core Function, EVOLVED-5G has developed a number of test cases to validate that after each deployment of CAPIF Core Function tool, API contracts are following 3GPP specifications (TS 23.222 [4], TS 29.222 [6] and TS 33.122 [6]) and that CAPIF Core Function behaves properly. These tests simulate *API Invoker* and *API Publisher* entities and invoke CAPIF APIs to test several conditions, checking that the response provided by CAPIF Core Function is the appropriate one.

CAPIF Core Function has published several releases along the EVOLVED-5G Project. Compared to the version used in D5.1, the latest release of the CAPIF Core Function has included additional APIs (such as Logging and Auditing API, and Security management) that were not implemented in the first releases and have added mutual TLS support to connections, which have significant impact in the implementation.

The defined tests, including new APIs added in CAPIF Core Function release 3.1.2, are summarized in the following table:

Table 20. Testing plan targeting the API Invoker and API Publisher of CAPIF

TEST	Entity	CAPIF API
Register NetApp	CAPIF_API_Invoker_Management_API	201 API invoker on-boarded successfully.
Register NetApp Already registered	CAPIF_API_Invoker_Management_API	403 Forbidden
Update Registered NetApp	CAPIF_API_Invoker_Management_API	200 API invoker details updated successfully.
Update Not Registered NetApp	CAPIF_API_Invoker_Management_API	404 Not found.
Delete Registered NetApp	CAPIF_API_Invoker_Management_API	204 The individual API Invoker matching onboardingId was offboarded.
Delete Not Registered NetApp	CAPIF_API_Invoker_Management_API	404 Not Found.
Register Api Provider	CAPIF_API_Provider_Management_API	201 Provider Registered
Register Api Provider Already registered	CAPIF_API_Provider_Management_API	403 Forbidden
Update Registered Api Provider	CAPIF_API_Provider_Management_API	200 OK API Provider Enrolment Details updated successfully
Update Not Registered Api Provider	CAPIF_API_Provider_Management_API	404 Not Found
Partially Update Registered Api Provider	CAPIF_API_Provider_Management_API	200 OK API Provider Enrolment Details updated successfully
Partially Update Not Registered Api Provider	CAPIF_API_Provider_Management_API	404 Not Found
Delete Registered Api Provider	CAPIF_API_Provider_Management_API	204 Individual Provider Deleted
Delete Not Registered Api Provider	CAPIF_API_Provider_Management_API	404 Not Found
Publish API by Authorised API Publisher	CAPIF_Publish_Service_API	201 API Published
Publish API by NON Authorised API Publisher	CAPIF_Publish_Service_API	401 Unauthorised
Retrieve all APIs Published by Authorised apfId	CAPIF_Publish_Service_API	200 Definition of all service API(s) published by the API publishing function.
Retrieve all APIs Published by NON Authorised apfId	CAPIF_Publish_Service_API	401 Unauthorized
Retrieve single APIs Published by Authorised apfId	CAPIF_Publish_Service_API	200 Definition of serviceApiId service API published by the API publishing function.

Retrieve single APIs non Published by Authorised apfId	CAPIF_Publish_Service_API	404 Not Found
Retrieve single APIs Published by NON Authorised apfId	CAPIF_Publish_Service_API	401 Unauthorized
Update API Published by Authorised apfId with valid serviceApiId	CAPIF_Publish_Service_API	200 Definition of service API updated successfully.
Update APIs Published by Authorised apfId with invalid serviceApiId	CAPIF_Publish_Service_API	404 Not Found
Update APIs Published by NON Authorised apfId	CAPIF_Publish_Service_API	401 Unauthorized
Delete API Published by Authorised apfId with valid serviceApiId	CAPIF_Publish_Service_API	204 The individual published service API matching the serviceAPIId is deleted.
Delete APIs Published by Authorised apfId with invalid serviceApiId	CAPIF_Publish_Service_API	404 Not Found
Delete APIs Published by NON Authorised apfId	CAPIF_Publish_Service_API	401 Unauthorized
Discover Published service APIs by Authorised API Invoker	CAPIF_Discover_Service_API	200 With Collection of Service API Descriptions
Discover Published service APIs by Non Authorised API Invoker	CAPIF_Discover_Service_API	401 Unauthorized
Discover Published service APIs by not registered API Invoker	CAPIF_Discover_Service_API	403 Forbidden
Discover Published service APIs by registered API Invoker with 1 result filtered	CAPIF_Discover_Service_API	200 Ok with 1 api returned
Discover Published service APIs by registered API Invoker filtered with no match	CAPIF_Discover_Service_API	200 Ok with empty list returned
Discover Published service APIs by registered API Invoker not filtered	CAPIF_Discover_Service_API	200 Ok with 2 api returned
Creates a new individual CAPIF Event Subscription	CAPIF_Events_Service_API	201 Subscription Created

Creates a new individual CAPIF Event Subscription with Invalid SubscriberId	CAPIF_Events_Service_API	404 Not Found
Deletes an individual CAPIF Event Subscription	CAPIF_Events_Service_API	204 Event subscription deleted
Deletes an individual CAPIF Event Subscription with invalid SubscriberId	CAPIF_Events_Service_API	404 Not Found
Deletes an individual CAPIF Event Subscription with invalid SubscriptionId	CAPIF_Events_Service_API	404 Not Found
Create a security context for an API invoker	CAPIF_Security_service_API	201 Security Context Created
Create a security context for an API invoker with Provider role	CAPIF_Security_service_API	401 Unauthorized
Create a security context for an API invoker with Provider entity role and invalid apiInvokerId	CAPIF_Security_service_API	401 Unauthorized
Create a security context for an API invoker with Invalid apiInvokerID	CAPIF_Security_service_API	404 Not Found
Retrieve the Security Context of an API Invoker	CAPIF_Security_service_API	200 OK with security service information.
Retrieve the Security Context of an API Invoker with invalid apiInvokerID	CAPIF_Security_service_API	404 Not Found
Retrieve the Security Context of an API Invoker with invalid apfId	CAPIF_Security_service_API	401 Unauthorized
Delete the Security Context of API Invoker	CAPIF_Security_service_API	204 Security context deleted
Delete the Security Context of API Invoker with Invoker entity role	CAPIF_Security_service_API	401 Unauthorized
Delete the Security Context of an API	CAPIF_Security_service_API	401 Unauthorized

Invoker with Invoker entity role and invalid apiInvokerID		
Delete the Security Context of an API Invoker with invalid apiInvokerID	CAPIF_Security_service_API	404 Not Found
Update the Security Context of an API Invoker	CAPIF_Security_service_API	200 OK Security service Information updated
Update the Security Context of an API Invoker with Provider entity role	CAPIF_Security_service_API	401 Unauthorized
Update the Security Context of an API Invoker with AEF entity role and invalid apiInvokerId	CAPIF_Security_service_API	401 Unauthorized
Update the Security Context of an API Invoker with invalid apiInvokerID	CAPIF_Security_service_API	404 Not Found
Revoke the authorization of the API invoker for APIs	CAPIF_Security_service_API	204 Revoked Authorization
Revoke the authorization of the API invoker for APIs without valid apfID.	CAPIF_Security_service_API	401 Unauthorized
Revoke the authorization of the API invoker for APIs with invalid apiInvokerId	CAPIF_Security_service_API	404 Not Found
Retrieve access token	CAPIF_Security_service_API	200 OK With access token information
Retrieve access token by Provider	CAPIF_Security_service_API	401 Unauthorized
Retrieve access token by Provider with invalid apiInvokerId	CAPIF_Security_service_API	401 Unauthorized
Retrieve access token with invalid apiInvokerId	CAPIF_Security_service_API	404 Not Found
Retrieve access token with invalid client_id	CAPIF_Security_service_API	400 Error
Retrieve access token with unsupported grant_type	CAPIF_Security_service_API	400 Error
Retrieve access token with invalid scope	CAPIF_Security_service_API	400 Error

Retrieve access token with invalid aefid at scope	CAPIF_Security_service_API	400 Error
Retrieve access token with invalid apiName at scope	CAPIF_Security_service_API	400 Error
Create a log entry	CAPIF_Loggin_Service_API	201 Log Entry Created
Create a log entry invalid aefId	CAPIF_Loggin_Service_API	404 Not Found
Create a log entry invalid serviceApi	CAPIF_Loggin_Service_API	404 Not Found
Create a log entry invalid apiInvokerId	CAPIF_Loggin_Service_API	404 Not Found
Create a log entry different aef_id in body	CAPIF_Loggin_Service_API	401 Unauthorized
Get Log Entry	CAPIF_Auditing_Service_API	200 OK With Log entry
Get a log entry without entry created	CAPIF_Auditing_Service_API	404 Not Found
Get a log entry withut aefid and apiInvokerId	CAPIF_Auditing_Service_API	400 Bad Request
Get Log Entry with apiVersion filter	CAPIF_Auditing_Service_API	200 OK With log filtered by apiVersion
Get Log Entry with no exist apiVersion filter	CAPIF_Auditing_Service_API	404 Not Found

All these tests have been implemented using Robot Framework tool [12] and are available at the EVOLVED-5G Github repository: https://github.com/EVOLVED-5G/CAPIF_API_Services, where different folders are created, one related to some aspect of CAPIF. This information was reported in D5.1 but it is replicated in this deliverable to facilitate the reader the completeness of the information. These folders are:

- Docs: Here all the documentation related to the Test Plan definition created for CAPIF is stored.
- Iac: This folder contains all needed information to deploy infrastructure of services at OpenShift, in this case, Terraform scripts.
- Pac: It contains pipelines to be used by Jenkins for any operation, like deploy/destroy at OpenShift, generation of Docker images for testing or launch test.
- Services: All services involved at CAPIF deployment, including auxiliary services like jwt, nginx, easyrsa server, etc.
- Tests: The Robot code for testing is under this folder, where Test Cases and all related code developed (like Python custom libraries and resources) are stored.
- Tools: This folder contains information to generate Robot Docker image (to be used mainly by Jenkins pipelines) and also script to generate from Swagger the initial template of CAPIF services.

The Tests folder contains all developed code of robot to execute Test Plan defined. Under tests folder a directory structure to split in a logical way all code needed is presented:

- Features:
    - Here are the Test Cases for each service.

  - Each folder (including root one) include a __init__.robot that setup configuration for all directories contained below it.
  - The code used is Robot flavor.
  - This code will use also code inside Libraries and Resources
- Libraries:
  - At this folder Python is used as an auxiliary code for testing.
  - This is a usual way to develop code needed for testing that need complex logic where Robot code is discouraged, because High level syntax only increase the complexity, for example, get an object from dictionary.
- Resources:
  - All auxiliary code developed using Robot, mainly Keywords and Variables for all Test Cases implementations.

Test Cases are split based on each component of the CAPIF. Each one will be stored under tests/features/<COMPONENT_NAME> folder at repository and will have 2 files:

- __init__.robot : This file contains all needed Settings for that specific component, for example, Force Tags that will setup the tag for all test of that component.

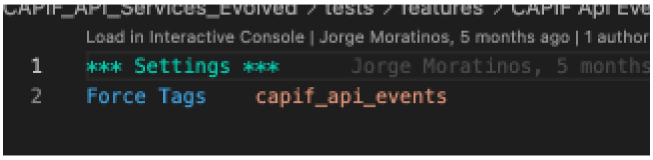

Figure 23. __init__.robot file

- <component_name>.robot : This file contains all code for each Test Case of that component, setting up the tags for each test.

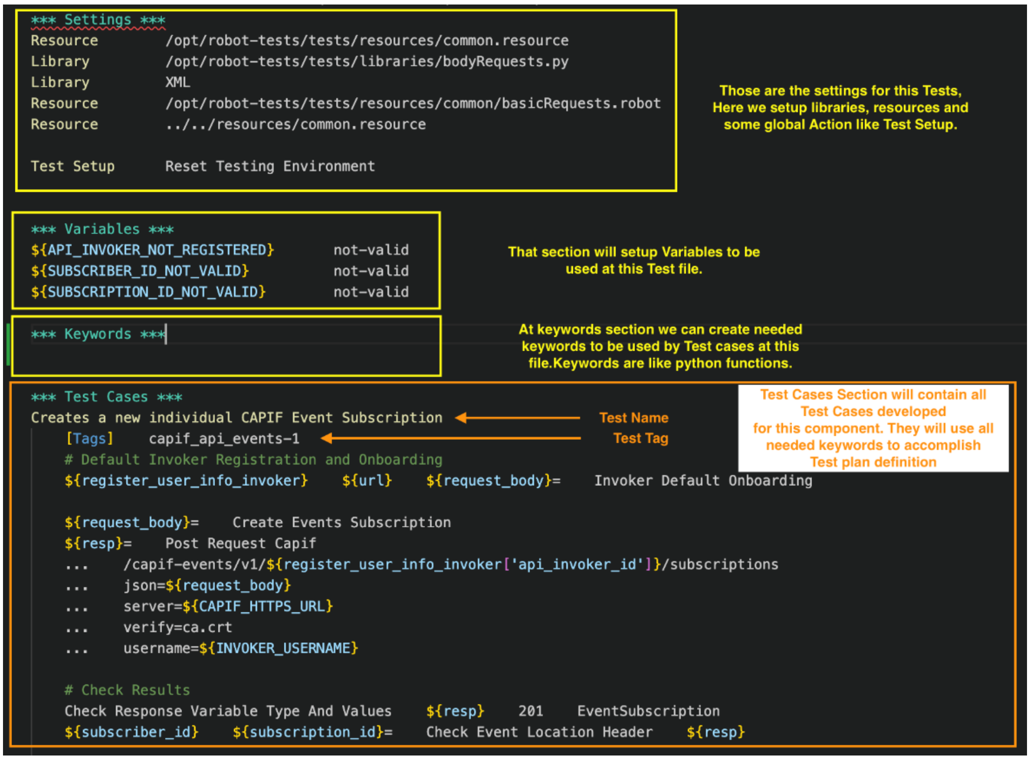

Figure 24. <component_name>.robot file

The __init_.robot file Force Tags simplifies the way tests are launched. Indeed, when launching all tests of one CAPIF component, the addition of those tags is the only necessary requirement.

For the generation of robot image, just the execution of the following command under repository folder /tools/robot (where Dockerfile is stored) is needed:

```
docker build –no-cache . -t ${ROBOT_IMAGE_NAME}:${ROBOT_VERSION}
```

This will build a Docker image with name and version indicated. For example, the command could be something like:

```
docker build –no-cache . -t dockerhub.hi.inet/5ghacking/5gnow-robot-test-image:3.0
```

To use Jenkins, the usual way is to build that and push this new image to dockerhub.hi.inet. The pipeline that manages that process is presented at the repository under /pac/Jenkinsfile-tools.groovy (https://github.com/EVOLVED-5G/CAPIF_API_Services/blob/develop/pac/Jenkinsfile-tools.groovy)

Once the image is built, there are 2 ways to execute Robot Tests:

- Local machine: This is useful during development, when a quick way of testing (either on local or remote environments) with CAPIF is enabled by using a local Docker image built previously.
    - This is allowed by building a local Docker image and launching it with the needed input parameters. (This is better for local development).
    - Alternatively, the image uploaded to dockerhub.hi.inet can also be downloaded.
- Jenkins: This is useful to raise up a complete ci/cd environment enabling the deployment of CAPIF services at OpenShift and launching tests on that deployment pipeline.
    - To allow this, the robot Docker image must be uploaded to Dockerhub, usually executing the tool build pipeline.

The way to invoke Docker image is the same, but in Jenkins a Groove pipeline is used instead. However, the step to launch tests using Docker image is the same:

```
docker run --tty --rm --network="host" \
    -v ${ROBOT_TESTS_DIRECTORY}:/opt/robot-tests/tests \
    -v ${ROBOT_RESULTS_DIRECTORY}:/opt/robot-tests/results ${ROBOT_IMAGE_NAME}:${ROBOT_VERSION}  \
    --variable CAPIF_HOSTNAME:${CAPIF_HOSTNAME} \
    --variable CAPIF_HTTP_PORT:${CAPIF_PORT} \
    --variable CAPIF_HTTPS_PORT:${CAPIF_TLS_PORT} \
    ${ROBOT_TESTS_INCLUDE} ${ROBOT_TEST_OPTIONS}
```

As seen in the above screenshot, the Docker command has 4 parts:

- “docker run -t –network=”host” –rm”: this will call Docker and set some useful options:
    - Run: launch Docker image on Docker.
    - -t: use tty
    - –network: this means Docker image will use the same networks than host.
    - –rm: this option will remove the image after end execution of tests from Docker environment.
- “-v <LOCAL_DIRECTORY>:<DOCKER_VOLUME>”: this option will attach local directory to volumes defined under robot Docker image. It only has 2 volumes:

  - /opt/robot-tests/tests: At this volume Docker image will expect to attach tests directory of repository, including all robot code.
  - /opt/robot-tests/results: At this volume robot will store reports generated after execution of tests. To get access to these reports, a folder on the host must be provided, otherwise these reports will be lost.
- "${ROBOT_IMAGE_NAME}:${ROBOT_VERSION}": This part indicates the Docker image and version previously generated that Docker will run.
- Options after Robot Image selection: The command after Docker image information will be sent as a part of robot command executed inside Docker. This means input variables can be placed:
  - --variable: This allows setting variables used by robot tests cases developed as input at call.
  - –include: This option sets tags to be executed where selected tests will execute robot.

To check if all CAPIF services are running properly in local machine after executing run.sh, the following command should be used:

./check_services_are_running.sh

This shell script will return 0 if all services are running properly.

After running all the tests, a report is produced gathering test results:

Tests Report

Generated
20230926 13:20:51 UTC+02:00
12 days 20 hours ago

Summary Information

Status:	All tests passed
Start Time:	20230926 13:08:31.476
End Time:	20230926 13:20:48.709
Elapsed Time:	00:12:17.233
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	76	76	0	0	00:12:16	

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Tests	76	76	0	0	00:12:17	
Tests . Features	76	76	0	0	00:12:17	
Tests . Features . CAPIF Api Auditing Service	5	5	0	0	00:01:03	
Tests . Features . CAPIF Api Auditing Service . Capif Auditing Api	5	5	0	0	00:01:03	
Tests . Features . CAPIF Api Discover Service	6	6	0	0	00:01:15	
Tests . Features . CAPIF Api Discover Service . Capif Api Service Discover	6	6	0	0	00:01:15	
Tests . Features . CAPIF Api Events	5	5	0	0	00:00:23	
Tests . Features . CAPIF Api Events . Capif Events Api	5	5	0	0	00:00:23	
Tests . Features . CAPIF Api Invoker Management	7	7	0	0	00:00:30	
Tests . Features . CAPIF Api Invoker Management . Capif Api Invoker Managenet	7	7	0	0	00:00:30	
Tests . Features . CAPIF Api Logging Service	5	5	0	0	00:01:03	
Tests . Features . CAPIF Api Logging Service . Capif Logging Api	5	5	0	0	00:01:03	
Tests . Features . CAPIF Api Provider Management	8	8	0	0	00:01:13	
Tests . Features . CAPIF Api Provider Management . Capif Api Provider Management	8	8	0	0	00:01:13	
Tests . Features . CAPIF Api Publish Service	13	13	0	0	00:02:08	
Tests . Features . CAPIF Api Publish Service . Capif Api Publish Service	13	13	0	0	00:02:08	
Tests . Features . CAPIF Security Api	27	27	0	0	00:04:41	
Tests . Features . CAPIF Security Api . Capif Security Api	27	27	0	0	00:04:41	

Figure 25. CAPIF Tool Results – Test Report

3.3 PERFORMANCE TESTING

The updated analysis of the performance of the NEF emulator and CAPIF services has been made by following the same methodology described in Deliverable D5.1, which consisted in the definition of a Test Case, presented below, designed for the collection of Mean Access Time metrics for all endpoints in the components. In order to allow comparison of the results, this Test Case, and the basic implementation is unchanged with respect to the one presented in the previous deliverable:

EVOLVED-5G Test Case Template	-ID number-	Generic Endpoint Performance Test	Access Time
Scenario (storyline)	*Description of the motivation and the scope of the test at a qualitative level. What is the reference scenario in a real (industrial) environment that we want to capture with this test?* The objective of the test is to measure the mean access time of the tested. The test also ensures that the endpoint is able to reliably provide the correct response.		
Testing Infrastructure (Pre-conditions)	*Information related to all the parameters that affect the values of the KPI/KVIs/KVIs to be measured, network deployment and environment conditions, etc. [Any requirement that needs to be done before execution of this test case. A list of test specific pre-conditions that need to be met by the SUT including information about equipment configuration, traffic descriptor]* • *The set of software and hardware components involved + their configuration* • *The service type + the traffic that is involved in the process* • The component that exposes the tested endpoint is running and listening for requests at a known address • The component that exposes the tested endpoint is prepared with the minimal set of configuration values required for the testing process		
Target KPI/KVI	*Here goes the definition of the target KPI/KVI. Each test case targets only one KPI/KVI (main KPI/KVI). However, secondary measurements from complementary KPI/KVIs can be added as well. The definition of the main KPI/KVI 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/KVI /KVI declares:* • *The definition of the KPI/KVI+ (if applicable) a secondary list of KPI/KVIs useful to interpret the values of the target KPI/KVI.* • *The reference points from which the measurement(s) will be performed* • *The reference protocol stack level where the measurement is performed* • *Target values + theoretical value space* The target KPI is the **Mean Access Time** of the tested endpoint, which is defined as endpoint the mean delay between the time in which a client sends a request to the tested endpoint till the time in which the client receives a correctly formed response to the request. In order to obtain a statistically meaningful result the endpoint is tested 100 times. The endpoint must provide a response with a delay that is below a certain threshold in all requests, in order to consider the test as Successful.		
Test Case Sequence	Specializes the measurement process (methodology) of the metric for the selected underlay system. In this field: • The steps to be followed for performing the measurements are specified • The iterations required, the monitoring frequency, etc., are declared.		

	○ The testing framework configures in the component all the required values that could not be prepared as part of the pre-conditions, such as creation of test users or entities, or checks if such configuration has already been performed by a previous test. ○ The following actions are repeated 100 times in order to obtain statistically meaningful results while also stressing the tested component: 2.1. The testing framework prepares any necessary data payloads for the tested endpoint, and/or makes use of other endpoints in order to prepare the component for receiving a request in the tested endpoint. 2.2. The testing framework sends a request to the tested endpoint, measuring the time required to receive a response. 2.3. The testing framework checks the received response: - If it is not well formatted or otherwise unexpected the test is finalized and considered **Failed.** - If the response is the expected, but the delay exceeds the defined threshold, the test is considered **Failed**, but continues in order to calculate a more accurate mean access time. 2.4. The measured delay is used to calculate the mean access time 2.5. If necessary, the testing framework performs any necessary cleanup before the next iteration starts. ○ Once all iterations have been completed (or an error has been detected): • If all iterations have been completed, and all the measured delays are below the defined threshold, then the test is considered **Successful**. • If all iterations have been completed, but any of them had a delay above the defined threshold, then the test is considered **Failed**. • If any of the iterations has not been completed (due to receiving an unexpected response or because of a runtime error), then the test is considered **Failed.**

This Test Case has been implemented using Robot Framework [10] , with a basic implementation of the timing measurement that is shared for both components (encapsulated as importable *keywords*) that are used as part of specific batteries of tests that are fine-tuned for each of the components.
All of the tests that are part of these campaigns follow the template displayed below, but are customized depending on the specific needs for testing each endpoint. For example, different endpoints may need some previous actions to be performed in order to prepare the environment or to clean-up leftovers from previous tests.

The following snippet shows the basic structure of all the tests, in ‘Example Endpoint Test’, as well as the main *keywords* used in the campaigns, ‘Handle Timing’ and ‘Handle End Results’:

```
*** Keywords ***

Handle Timing
    [Arguments]   ${elapsed}  ${iteration}      ${average}  ${success}
    ${timespan}   Evaluate \
            ${elapsed.seconds}+(${elapsed.microseconds}/1000000.0)
    Log To Console      <<<[${TEST_NAME}]${iteration}=${timespan}>>>
    IF  ${timespan} < ${THRESHOLD}
        Log To Console  Success
        ${success}  Evaluate    ${success}+${1}
    ELSE
        Log To Console  Fail
    END
    IF  ${iteration} < ${1}
        ${average}=     Set Variable    ${timespan}
```

```
    ELSE
        ${average}=     Evaluate   \
            ((${average}*(${iteration}))+${timespan})/${iteration+1}
    END
    [Return]    ${success}      ${average}

Handle End Results
    [Arguments]     ${success}      ${average}
    Log To Console     \
      <<<[${TEST_NAME}];Success=${success}/${ITERATIONS};Average=${average}>>>
    IF      ${success} < ${ITERATIONS}
        Fail    Detected response times above threshold
    END


*** Test Cases ***

Example Endpoint Test
    # Prepare general variables for the test
    ${success}=     Set Variable    ${0}
    ${average}=     Set Variable    ${0}

    FOR     ${index}    IN RANGE    ${ITERATIONS}
        Log To Console      Iteration: ${index}

        # Prepare any required payloads or make use of additional
        # endpoints to prepare the component (step 2.1)

        # Step 2.2
        ${resp}=    GET    <endpoint>    headers=${header}

        # Handle Timing performs the required calculations (steps 2.3
        # and 2.4)
        ${success}  ${average}  Handle Timing \
            ${resp.elapsed}  ${index}  ${average}  ${success}

        # Any cleanup necessary before the next iteration
        # is performed here (step 2.5)
    END

    Handle End Results      ${success}      ${average}
```

Figure 26. Common performance tests implementation

The following sub-sections describe the results obtained during the execution of the performance tests campaigns for the NEF Emulator and the CAPIF services.

3.3.1 NEF Emulator

The test campaign defined for the NEF Emulator covers all the functionality provided by the component. The tests included per API are:

- Cells API:
  - Create valid cell valid token
  - Delete valid cell valid token
  - Read valid cell valid token
  - Read cell valid gnb valid token
  - Read cells valid token
  - Update valid cell valid token

- Web API:
  - Authorized access dashboard page
  - Authorized access map page
  - Authorized access import page
  - Authorized access export page

- gNB API:
  - Create valid gNB valid token
  - Delete valid gNB valid token
  - Read by id valid gNB valid token
  - Read gNBs valid token
  - Update valid gNB valid token

- Login API:
  - Get valid access token
  - Test valid access token

- Movement API:
  - Initiate movement valid token
  - Terminate movement valid token
  - State UEs valid token

- Monitoring Events API:
  - List Active Event Subscription Performance
  - Event Subscription Creation Performance
  - Event Subscription Read Performance
  - Event Subscription Update Performance
  - Event Subscription Delete Performance

- Paths API:
  - Create valid path valid token
  - Delete valid path valid token
  - Read valid path valid token
  - Read paths valid token
  - Update valid path valid token

- QoS Information API:
  - Get QoS characteristics valid token

- UE API:
  - Assign predefined path valid UE valid token
  - Create valid UE valid token
  - Delete valid UE valid token
  - Read valid UE valid token
  - Read UEs valid token
  - Update valid UE valid token

- Session with QoS API:
  - List Active QoS Subscription Performance
  - QoS Subscription Creation Performance

  - QoS Subscription Read Performance
  - QoS Subscription Update Performance
  - QoS Subscription Delete Performance

- UI API:
  - Create valid monitoring callback
  - Create valid QoS callback
  - Export scenario valid token
  - Get notifications valid token
  - Import valid scenario valid token

- Users API:
  - Create valid user valid token
  - Create valid user without token
  - Read By Id valid user valid token
  - Read user 'me' valid token
  - Read users valid token
  - Update valid user valid token
  - Update 'me' valid token

Table 21 shows the results obtained by the NEF Emulator. Compared with the results obtained in the initial testing (in that case only some of these APIs where tested), the results are very similar and within the same timing range. The results demonstrate that the API endpoints are very fast in most cases and should not be the cause for any bottleneck in the Network Applications using the NEF Emulator.

Table 21. NEF Emulator performance test result.

API	Test	Average access time (S)	Success ratio
Cells API	Create valid cell valid token	0.026	100% - Success
	Delete valid cell valid token	0.031	100% - Success
	Read valid cell valid token	0.023	100% - Success
	Read cell valid gNB valid token	0.023	100% - Success
	Read cells valid toke	0.021	100% - Success
	Update valid cell valid token	0.031	100% - Success
Web API	Authorized access dashboard page	0.032	100% - Success
	Authorized access map page	0.029	100% - Success
	Authorized access import page	0.028	100% - Success
	Authorized access export page	0.034	100% - Success
gNB API	Create valid gNB valid token	0.023	100% - Success
	Delete valid gNB valid token	0.028	100% - Success
	Read by id valid gNB valid token	0.015	100% - Success
	Read gNBs valid token	0.015	100% - Success
	Update valid gNB valid token	0.024	100% - Success

Login API	Get valid access token	0.310	100% - Success
	Test valid access token	0.011	100% - Success
Movement API	Initiate movement valid token	0.020	100% - Success
	Terminate movement valid token	1.008	100% - Success
	State UEs valid token	0.009	100% - Success
Monitoring Events API	List Active Event Subscription	0.009	100% - Success
	Event Subscription Creation	0.015	100% - Success
	Event Subscription Read	0.009	100% - Success
	Event Subscription Update	0.011	100% - Success
	Event Subscription Delete	0.010	100% - Success
Paths API	Create valid path valid token	0.032	100% - Success
	Delete valid path valid token	0.034	100% - Success
	Read valid path valid token	0.017	100% - Success
	Read paths valid token	0.016	100% - Success
	Update valid path valid token	0.020	100% - Success
QoS Information API	Get QoS characteristics valid token	0.011	100% - Success
UE API	Assign predefined path valid UE valid token	0.017	100% - Success
	Create valid UE valid token	0.030	100% - Success
	Delete valid UE valid token	0.022	100% - Success
	Read valid UE valid token	0.016	100% - Success
	Read UEs valid token	0.017	100% - Success
	Update valid UE valid token	0.024	100% - Success
Session with QoS API	List Active QoS Subscription	0.036	100% - Success
	QoS Subscription Creation	0.043	100% - Success
	QoS Subscription Read	0.037	100% - Success
	QoS Subscription Update	0.039	100% - Success
	QoS Subscription Delete	0.037	100% - Success
UI API	Create valid monitoring callback	0.009	100% - Success
	Create valid QoS callback	0.010	100% - Success
	Export scenario valid token	0.135	100% - Success
	Get notifications valid token	0.014	100% - Success
	Import valid scenario valid token	0.772	100% - Success
Users API	Create valid user valid token	3.080	100% - Success
	Create valid user without token	0.294	100% - Success

	Read By Id valid user valid token	0.011	100% - Success
	Read user ‘me’ valid token	0.014	100% - Success
	Read users valid token	0.015	100% - Success
	Update valid user valid token	0.296	100% - Success
	Update ‘me’ valid token	0.297	100% - Success

3.3.2 CAPIF Tool

As in the case of NEF, all the functionality provided by the CAPIF services has been tested as part of the performance testing campaign, including all the new functionality and APIs implemented after the initial performance assessment:

- Auditing API:
    - Get log entry
    - Get log entry without an entry created
- API service discover:
    - Discover Published service APIs by Authorized API Invoker
    - Discover Published service APIs by registered API Invoker with 1 result filtered
    - Discover Published service APIs by registered API Invoker filtered with no match
    - Discover Published service APIs by registered API Invoker not filtered
- Events API:
    - Creates a new individual CAPIF Event Subscription
    - Deletes an individual CAPIF Event Subscription
- API Invoker Management:
    - Onboard NetApp
    - Update Onboarded NetApp
    - Offboard NetApp
    - Update Onboarded NetApp Certificate
    - Update Onboarded NetApp Notification Destination
- Logging API:
    - Create a log entry
- API Provider Management:
    - Register API Provider
    - Update Registered API Provider
    - Partially Update Registered API Provider
    - Delete Registered API Provider
- Service API publishing:
    - Publish API by Authorized API Publisher
    - Retrieve all APIs Published by Authorized apfId
    - Retrieve single APIs Published by Authorized apfId
    - Retrieve single APIs non-Published by Authorized apfId
    - Update API Published by Authorized apfId with valid serviceApiId

- Delete API Published by Authorized apfId with valid serviceApiId

- Security API:
  - Create a security context for an API invoker
  - Retrieve the Security Context of an API Invoker
  - Delete the Security Context of an API Invoker
  - Update the Security Context of an API Invoker
  - Revoke the authorization of the API invoker for APIs
  - Retrieve access token

Table 21 shows the results obtained for CAPIF Services. In this case we can observe a steep increase in the mean response time for all of the APIs when compared with the results reported in Deliverable 5.1. This is due to the overhead caused by the introduction of mutual Transport Layer Security (TLS) authentication, which is mandatory according to 3GPP specification TS 33.122, implemented in a CAPIF Core Tool released after the execution of the initial tests, however, given that most interactions with the CAPIF services are performed as administrative one-time processes these should not noticeably burden the performance of compatible Network Applications.

Table 22. CAPIF Tool performance test result.

API	Test	Average access time (S)	Success ratio
Auditing API	Get log entry	1.567	100% - Success
	Get log entry without an entry created	1.566	100% - Success
API service discover	Discover Published service APIs by Authorized API Invoker	1.553	100% - Success
	Discover Published service APIs by registered API Invoker with 1 result filtered	1.529	100% - Success
	Discover Published service APIs by registered API Invoker filtered with no match	1.565	100% - Success
	Discover Published service APIs by registered API Invoker not filtered	1.567	100% - Success
Events API	Creates a new individual CAPIF Event Subscription	1.508	100% - Success
	Deletes an individual CAPIF Event Subscription	1.521	100% - Success
API Invoker Management	Onboard NetApp	1.722	100% - Success
	Update Onboarded NetApp	1.604	100% - Success
	Offboard NetApp	1.527	100% - Success
	Update Onboarded NetApp Certificate	1.702	100% - Success
	Update Onboarded NetApp Notification Destination	1.598	100% - Success
Logging API	Create a log entry	1.650	100% - Success
API Provider Management	Register API Provider	1.937	100% - Success
	Update Registered API Provider	2.000	100% - Success

	Partially Update Registered API Provider	1.499	100% - Success
	Delete Registered API Provider	1.481	100% - Success
Service API publishing	Publish API by Authorized API Publisher	1.506	100% - Success
	Retrieve all APIs Published by Authorized apfId	1.555	100% - Success
	Retrieve single APIs Published by Authorized apfId	1.550	100% - Success
	Retrieve single APIs non-Published by Authorized apfId	1.529	100% - Success
	Update API Published by Authorized apfId with valid serviceApiId	1.574	100% - Success
	Delete API Published by Authorized apfId with valid serviceApiId	1.542	100% - Success
Security API	Create a security context for an API invoker	1.521	100% - Success
	Retrieve the Security Context of an API Invoker	1.533	100% - Success
	Delete the Security Context of an API Invoker	1.521	100% - Success
	Update the Security Context of an API Invoker	1.546	100% - Success
	Revoke the authorization of the API invoker for APIs	1.544	100% - Success
	Retrieve access token	1.528	100% - Success

4 CONCLUSION

This deliverable presents the results of the final evaluation trials of the EVOLVED-5G platforms, and the software components developed in the context of the project, providing a view on the status and performance of the infrastructure after the conclusion of the second phase of the project.

With regards to the performance measured in the different EVOLVED-5G sites, the results obtained have been improved reaching throughputs of up to 1,3 Gbps and latencies lower than 10 ms for mmWave configurations. The results obtained when using the TSN deployment on the Málaga platform have also been improved obtaining one-way delays of 5ms for frequency range 1.

Functional tests have been extended to cover the new development done regarding CAPIF and NEF standards. With regards to the performance of the software components, the NEF Emulator and CAPIF Tool have achieved outstanding results, with a 100% success rate in all tests. The access times have been increased due to the addition of mutual TLS authentication and the results are around 1 second.

This deliverable demonstrates that Málaga and Athens platforms are valid to be used for the Network App Validation and Certification tests that will be explained in the deliverables 5.5 and 5.6. Both, the 5G infrastructure and the 5G components developed within the project performed according to the expectations and the Success Rate achieved (100%) demonstrate that the environments are stable and ready to be used for Network Applications.

5 REFERENCES

[1] A. Díaz Zayas, G. Caso, Ö. Alay, P. Merino, A. Brunstrom, D. Tsolkas, and H. Koumaras, "A Modular Experimentation Methodology for 5G Deployments: The 5GENESIS Approach," Sensors, vol. 20, no. 22, p. 6652, Nov. 2020 [Online]: http://dx.doi.org/10.3390/s20226652

[2] EVOLVED-5G D2.2 - Design of NetApps development and evaluation environments [Online]: https://evolved-5g.eu/wp-content/uploads/2021/11/EVOLVED-5G-D2.2-v1.0_final.pdf

[3] 3GPP TS 38.306 (V16.7.0), User Equipment (UE) radio access capabilities, Release 16, December 2021 https://www.etsi.org/deliver/etsi_ts/138300_138399/138306/16.03.00_60/ts_138306v160300p.pdf

[4] TS 29.522 "5G System; Network Exposure Function Northbound APIs"; Stage 3; v18.1.0, 2023 [Online]: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3337, Retrieve 07/22

[5] 3GPP TS 23.222 Common API framework for 3GPP Northbound APIs. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3450

[6] 3GPP TS 29.222 Common API Framework for 3GPP Northbound APIs [Online]: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3450

[7] 3GPP TS 33.122 Security aspects of Common API Framework (CAPIF) for 3GPP northbound APIs : https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3420

[8] EVOLVED-5G, "D4.2 EVOLVED-5G Factory of the Future (FoF) NetApps" [Online]: https://evolved-5g.eu/wp-content/uploads/2022/06/EVOLVED-5G-D4.2_v_1.0.pdf

[9] EVOLVED-5G, "D3.1 Implementations and integrations towards EVOLVED-5G framework realisation" : https://evolved-5g.eu/wp-content/uploads/2022/01/EVOLVED-5G-D3.1-v1.0.pdf

[10] EVOLVED-5G, Deliverable D3.2 "NetApp Certification Tools and Marketplace development(intermediate)": https://evolved-5g.eu/wp-content/uploads/2022/07/EVOLVED-5G-D3.2_FINAL.pdf

[11] EVOLVED-5G, "D4.1 5G Exposure Capabilities for Vertical Applications (Intermediate)" [Online]: https://evolved-5g.eu/wp-content/uploads/2022/03/EVOLVED-5G-D4.1_v2.0-final.pdf

[12] Robot Framework – Homepage [Online]: https://robotframework.org

[13] CAPIF standard specification YAMLs [Online]: https://github.com/jdegre/5GC_APIs.

[14] Nemo Outdoor 5G NR Drive Test Solution: https://www.keysight.com/es/en/product/NTA00002B/nemo-outdoor-5g-nr-drive-test-solution.html

[15] InfluxData – Homepage [Online]: https://www.influxdata.com/

[16] Grafana Labs – Homepage [Online]: https://grafana.com/

[17] Programming Protocol-independent Packet Processors (P4) Open Source Programming Language – Homepage [Online]: https://p4.org/

[18] 5GENESIS, "D2.4 Final report on facility design and experimentation planning": https://5genesis.eu/wp-content/uploads/2020/07/5GENESIS_D2.4_v1.0.pdf

[19] IEEE 802.1 Time-Sensitive Networking Task Group – Homepage: https://www.ieee802.org/1/pages/tsn.html

[20] 5G-ACIA, 5G Alliance for Connected Industries and Automation, White Paper 5G for Connected Industries and Automation Second Edition, February 2019.

[21] 5Genesis, "D5.2 System-Level Tests and Verification (Release B): https://5genesis.eu/wp-content/uploads/2021/10/5GENESIS_D5.2_v1.0.pdf

[22] 5Genesis, "D5.4 Documentation and supporting material for 5G stakeholders (Release B)" https://5genesis.eu/wp-content/uploads/2021/08/5GENESIS-D5.4_v1.0.pdf

[23]5G PPP Trials Results 2022 - Key Performance Indicators measured in advanced 5G Trial Sites, https://5g-ppp.eu/wp-content/uploads/2023/06/5GPPP-TMV_White-paper_Final-Trial-KPIs_v1.0.pdf

[24] EVOLVED-5G, Deliverable D3.3 "Implementations and integrations towards EVOLVED-5G framework realisation (final) https://evolved-5g.eu/wp-content/uploads/2023/05/EVOLVED-5G-D3.3_FV.pdf

6 ANNEXES

6.1 TEST CASE TEMPLATE

EVOLVED-5G Test Case Template	-ID number-	-Title-	- Target Metric (KPI family) -
Scenario (storyline)	*Description of the motivation and the scope of the test in a qualitative level. What is the reference scenario in a real (industrial) environment that we want to capture with this test?*		
Testing Infrastructure (Pre-conditions)	*Information related to all the parameters that affect the values of the KPIs to be measured, network deployment and environment conditions, etc. [Any requirement that needs to be done before execution of this test case. A list of test specific pre-conditions that need to be met by the SUT including information about equipment configuration, traffic descriptor]* • *The set of software and hardware components involved + their configuration* • *The service type + the traffic that is involved in the process*		
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. 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:* • *The definition of the KPI + (if applicable) a secondary list of KPIs useful to interpret the values of the target KPI.* • *The reference points from which the measurement(s) will be performed* • *The reference protocol stack level where the measurement is performed* • *Target values + theoretical value space*		
Test Case Sequence	Specializes the measurement process (methodology) of the metric for the selected underlay system. In this field: • The steps to be followed for performing the measurements are specified • The iterations required, the monitoring frequency, etc., are declared.		

6.2 Athens Platform Test Case Templates

(To be updated)

6.2.1 DL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	-NCSRD_Downlink-	-DL Throughput-	- Throughput (Mbps) -
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theorical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • OnePlus 8 PRO 5G (COTS UE) • Amarisoft RAN (5G NR Rel. 16) • Amarisoft Core (5GC Rel. 16) • Dell Laptop Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe **Pre-conditions:** 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 632628, 3489.42 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols • 256QAM modulation in DL • 2x2 MIMO layers		
Target KPI	The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is TCP. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below: $$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$$		

Test Case Sequence	1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start UMA iPerf android application in server mode 4. Instructing VM iPerf probe to generate traffic towards the UE 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (UE), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards

6.2.2 UL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	- NCSRD_uplink -	-UL Throughput-	- Throughput (Mbps) -
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theorical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • OnePlus 8 PRO 5G (COTS UE) • Amarisoft RAN (5G NR Rel. 16) • Amarisoft Core (5GC Rel. 16) • Dell Laptop Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe **Pre-conditions:** 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 632628, 3489.42 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols • 256QAM modulation in UL • SISO layer		

Target KPI	The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is TCP. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below: $$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$$
Test Case Sequence	1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start iPerf probe on VM in server mode 4. Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (VM), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards

6.2.3 Best UL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	**- NCSRD_best_uplink -**	**-UL Throughput-**	**- Throughput (Mbps) -**
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theorical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.		

Testing Infrastructure (Pre-conditions)

Hardware and Software components:

Hardware components
- OnePlus 8 PRO 5G (COTS UE)
- Amarisoft RAN (5G NR Rel. 16)
- Amarisoft Core (5GC Rel. 16)
- Dell Laptop

Software components
- UMA iPerf (Android Application)
- OpenTAP for automated testing (iPerf TAP plugin)
- Open5Genesis iPerf probe

Pre-conditions:
1. VM with Open5Genesis iPerf probe is up and running
2. COTS UE has 5G connectivity
3. UMA iPerf application is installed on the COTS UE
4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP)

Radio Configuration:
- n78 band
- ARFCN 632628, 3489.42 MHz
- 50 MHz channel bandwidth
- 30 KHz SCS
- TDD, 1 DL slots, 8 UL slots, 1 special slot with 10 DL symbols and 2 UL symbols
- 256QAM modulation in UL
- SISO layer

Target KPI

The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is TCP. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:

$$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$$

Test Case Sequence

1. Instantiation of the slice, deployment of VM running the iPerf probe
2. Run script to ensure that the service on VM is running
3. Start iPerf probe on VM in server mode
4. Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM
5. Stop iPerf probe on VM
6. Retrieve experiment results from the iPerf server (VM),
7. Extract KPIs and persist to database
8. Iterate steps 3-7 three times
9. Generate statistical analysis and graphical timeline dashboards

6.2.4 DL throughput (COSMOTE)

EVOLVED-5G Test Case Template	-COS_Downlink-	-DL Throughput-	- Throughput (Mbps) -
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the COSMOTE site. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Samsung Galaxy S20 5G (COTS UE) • Ericsson BB6630, Radio4408 ((3GPP TS 37.141 version 16.6.0 Release 16) • Athonet 5G Core (SA) • Dell Laptop Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe **Pre-conditions:** 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 636666, 3500 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD • 256QAM modulation in DL, • 4x4 MIMO		
Target KPI	The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below: $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$		

Test Case Sequence	1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start UMA iPerf android application in server mode 4. Instructing VM iPerf probe to generate traffic towards the UE 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (UE), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards

6.2.5 UL throughput (Cosmote)

EVOLVED-5G Test Case Template	**- COS_uplink -**	**-UL Throughput-**	**- Throughput (Mbps) -**
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the COSMOTE site. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Samsung Galaxy S20 5G (COTS UE) • Ericsson BB6630, Radio4408 (3GPP TS37.141 version 16.6.0 Release 16)Athonet 5G Core • Dell Laptop Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe **Pre-conditions:** 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 636666, 3500 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD • 64QAM modulation in UL • SISO		

Target KPI	The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below: $$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$$
Test Case Sequence	1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start iPerf probe on VM in server mode 4. Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (VM), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards

6.2.6 RTT (NCSRD Demokritos)

EVOLVED-5G Test Case Template	**-NCSRD_RTT-**	**-RTT-**	**- Delay (ms) -**
Scenario (storyline)	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site)		

<table>
<tr><td>Testing Infrastructure (Pre-conditions)</td><td>Hardware and Software components:

Hardware components
• OnePlus 8 PRO 5G (COTS UE)
• Amarisoft RAN (5G NR Rel. 16)
• Amarisoft Core (5GC Rel. 16)
• Dell Laptop

Software components
• OpenTAP for automated testing (ping TAP plugin)
• Open5Genesis ping probe

Pre-conditions:
1. VM with Open5Genesis ping probe is up and running
2. COTS UE has 5G connectivity
3. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP)

Radio Configuration:
• n78 band
• ARFCN 632628, 3489.42 MHz
• 50 MHz channel bandwidth
• 30 KHz SCS
• TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols
• 256QAM modulation in UL
• SISO layer
• 20 ms scheduling request period</td></tr>
<tr><td>Target KPI</td><td>The target KPI of this test is to measure the RTT in ms. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.</td></tr>
<tr><td>Test Case Sequence</td><td>1. Instantiation of the slice, deployment of VM running the ping probe
2. Run script to ensure that the service on VM is running
3. Instructing ping probe (VM) to send ICMP echo requests to the target UE
4. Stop ping probe on VM
5. Retrieve experiment results from the ping server (VM),
6. Extract KPIs and persist to database
7. Iterate steps 3-7 three times
8. Generate statistical analysis and graphical timeline dashboards</td></tr>
</table>

6.2.7 RTT low latency (NCSRD Demokritos)

<table>
<tr><td>EVOLVED-5G Test Case Template</td><td>- NCSRD_RTT_low_latency-</td><td>-RTT-</td><td>- Delay (ms) -</td></tr>
<tr><td>Scenario (storyline)</td><td colspan="3">This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site)</td></tr>
</table>

Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • OnePlus 8 PRO 5G (COTS UE) • Amarisoft RAN (5G NR Rel. 16) • Amarisoft Core (5GC Rel. 16) • Dell Laptop Software components • OpenTAP for automated testing (ping TAP plugin) • Open5Genesis ping probe **Pre-conditions:** 1. VM with Open5Genesis ping probe is up and running 2. COTS UE has 5G connectivity 3. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 632628, 3489.42 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, 2 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols (i.e., in 2.5 ms) • 256QAM modulation in UL • SISO layer • 0.5 ms scheduling request period
Target KPI	The target KPI of this test is to measure the RTT in ms. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.
Test Case Sequence	1. Instantiation of the slice, deployment of VM running the ping probe 2. Run script to ensure that the service on VM is running 3. Instructing ping probe (VM) to send ICMP echo requests to the target UE 4. Stop ping probe on VM 5. Retrieve experiment results from the ping server (VM), 6. Extract KPIs and persist to database 7. Iterate steps 3-7 three times 8. Generate statistical analysis and graphical timeline dashboards

6.2.8 RTT (Cosmote)

EVOLVED-5G Test Case Template	**-COSMOTE_RTT-**	**-RTT-**	**- Delay (ms) -**
Scenario (storyline)	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform (i.e., COSMOTE site)		

Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Samsung Galaxy S20 5G (COTS UE) • Ericsson BB6630, Radio 4408 (3GPP TS 37.141 version 16.6.0 Release 16)Athonet 5G Core • Dell Laptop Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (ping TAP plugin) • Open5Genesis ping probe **Pre-conditions:** 1. VM with Open5Genesis ping probe is up and running 2. COTS UE has 5G connectivity 3. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) **Radio Configuration:** • n78 band • ARFCN 636666, 3500 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD • 256QAM modulation in DL, 64QAM modulation in UL • 4x4 MIMO
Target KPI	The target KPI of this test is to measure the RTT in ms. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.
Test Case Sequence	1. Instantiation of the slice, deployment of VM running the ping probe 2. Run script to ensure that the service on VM is running 3. Instructing ping probe (VM) to send ICMP echo requests to the target UE 4. Stop ping probe on VM 5. Retrieve experiment results from the ping server (VM), 6. Extract KPIs and persist to database 7. Iterate steps 3-7 three times 8. Generate statistical analysis and graphical timeline dashboards

6.3 UMA PLATFORM TEST CASE TEMPLATES

6.3.1 DL throughput (UMA)

EVOLVED-5G Test Case Template	-UMA_Downlink-	-DL Throughput-	- Throughput (Mbps) -
Scenario (storyline)	This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the **throughput** of the 5G infrastructure that lays on the UMA platform and compare the results with theorical values.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • One plus 9 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) Software components • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe **Pre-conditions:** 5. VM with Open5Genesis iPerf probe is up and running 6. COTS UE has 5G connectivity 7. UMA iPerf application is installed on the COTS UE 8. Network is configured **Radio Configuration:** • n78 band • ARFCN 651666, 3774.990 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 4x4 DL MIMO layers		
Target KPI	The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below: $$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^{J} \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot \left(1 - OH^{(j)}\right) \right)$$		

Test Case Sequence	1. Start UMA iPerf android application in server mode 2. Start iPerf client to generate traffic towards the UE 3. Stop iPerf probe on VM 4. Retrieve experiment results from the iPerf server (UE), 5. Extract KPIs and persist to database 6. Iterate steps 2-4 25 times 7. Generate statistical analysis and graphical timeline dashboards

6.3.2 RTT (UMA)

EVOLVED-5G Test Case Template	**-UMA_RTT-**	**-RTT-**	**- Delay (ms) -**
Scenario (storyline)	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the UMA platform.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Samsung Galaxy S20 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) Software components • OpenTAP for automated testing (ping TAP plugin) • Open5Genesis ping probe **Pre-conditions:** 4. VM with Open5Genesis ping probe is up and running 5. COTS UE has 5G connectivity 6. Network is configured **Radio Configuration:** • n78 band • ARFCN 651666, 3774.990 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 4x4 DL MIMO layers		
Target KPI	The target KPI of this test is to measure the RTT in ms. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer.		

Test Case Sequence	1. Start ping probe install at the UE to send ICMP echo requests to the main compute node 2. Stop ping probe on UE 3. Retrieve experiment results from the ping UE 4. Extract KPIs and persist to database 5. Iterate steps 1-3 three times 6. Generate statistical analysis and graphical timeline dashboards

6.3.3 TSN One-Way delay (UMA)

EVOLVED-5G Test Case Template	**-UMA_TSN_OWD-**	**-TSN_OWD-**	**- Delay (ms) -**
Scenario (storyline)	This test evaluates the One-Way Delay (OWD) of a TSN over 5G SA network. The main goal of this test is to assess the end-to-end delay of the TSN over 5G infrastructure that lays on the UMA platform.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Telit fn980m (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Open5GS Core (5GC) Software components • packetETH (Traffic generator) • UMA monitoring tool **Pre-conditions:** 1. TSN equipment synchronized 2. TSN P4 translators up and running 3. COTS UE has 5G connectivity 4. Network is configured 5. Monitoring tool running at TSN endpoints **Radio Configuration:** • n78 band • ARFCN 651666, 3774.990 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 4x4 DL MIMO layers		
Target KPI	The target KPI of this test is to measure the OWD in nsec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since a traffic generator is used to configure packets on demand, a TSN traffic profile has been used and a tool has been developed to measure and calculate different KPIs of the connection.		

Test Case Sequence	1. Start sending customized UDP packets from one TSN endpoint to the other TSN endpoint 2. Stop sending traffic after 1h 3. Experimental results are processed automatically in real-time 4. Retrieve the processed KPIs from the database 5. Iterate steps 1-4 (at least) three times 6. Generate statistical analysis and graphical timeline dashboards

6.3.4 TSN Jitter (UMA)

EVOLVED-5G Test Case Template	-UMA_TSN_Jitter-	-TSN_Jitter-	- Jitter (ms) -
Scenario (storyline)	This test evaluates the Jitter of a TSN over 5G SA network. The main goal of this test is to assess the end-to-end jitter of the TSN over 5G infrastructure that lays on the UMA platform.		
Testing Infrastructure (Pre-conditions)	**Hardware and Software components:** Hardware components • Telit fn980m (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Open5GS Core (5GC) Software components • packetETH (Traffic generator) • UMA monitoring tool **Pre-conditions:** 1. TSN equipment synchronized 2. TSN P4 translators up and running 3. COTS UE has 5G connectivity 4. Network is configured 5. Monitoring tool running at TSN endpoints **Radio Configuration:** • n78 band • ARFCN 651666, 3774.990 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 4x4 DL MIMO layers		
Target KPI	The target KPI of this test is to measure the jitter in nsec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since a traffic generator is used to configure packets on demand, a TSN traffic profile has been used and a tool has been developed to measure and calculate different KPIs of the connection.		
Test Case Sequence	1. Start sending customized UDP packets from one TSN endpoint to the other TSN endpoint 2. Stop sending traffic after 1h 3. Experimental results are processed automatically in real-time 4. Retrieve the processed KPIs from the database 5. Iterate steps 1-4 (at least) three times 6. Generate statistical analysis and graphical timeline dashboards		