Grant Agreement No: 101016941

Full Title:	Open cooperative 5G experimentation platforms for the industrial sector NetApps
Start date:	01/01/2021
End date:	30/06/2024
Duration:	42 Months
Project URL:	https://www.5g-induce.eu/

Deliverable D6.1

Testbed structure, validation and performance estimation outcomes

Document type	Deliverable		
Title	D6.1 – Testbed structure, validation and performance estimation outcomes		
Contractual due date	30/06/2024 (M42)	**Actual submission date**	30/08/2024
Nature	Report	**Dissemination Level**	Public
Lead Beneficiary	CNIT		
Responsible Author	Chiara Lombardo (CNIT)		
Contributions from	Chiara Lombardo (CNIT), Roberto Bruschi (CNIT), Franco Davoli (CNIT), Dimitrios Klonidis (UBI), Thanos Xirofotos (UBI), Christina Lessi (OTE), Diego San Cristobal Elpaza (ERC), Enrica Bosani (WHR), Cristiano Uboldi (WHR), Guerino Lamanna (INFO), Maurizio Giribaldi (INFO)		

Revision history

Version	Issue Date	Changes	Contributor(s)
v0.1	31/10/2023	Initial version	Chiara Lombardo (CNIT)
v0.2	11/12/2023	Spanish Experimental Facility	ERC, FORD
v0.3	21/05/2024	Revision and updated structure	CNIT, UBI
V0.4	28/07/2024	Italian Experimental Facility	WHR
V0.5	24/08/2024	Greek Experimental Facility	OTE
v1.0	30/08/2024	Final revision – Version ready for submission	Franco Davoli (CNIT)

Disclaimer

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the 5G-INDUCE consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the 5G-INDUCE Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing neither the 5G-INDUCE Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.

Copyright message

Table of Contents

List of Figures

List of Tables

Glossary of terms and abbreviations used

Abbreviation / Term	Description
eMBB	enhanced Mobile BroadBand
mMTC	massive Machine Type Communications
URLLC	Ultra-Reliable and Low Latency Communications
CVM	Core Validation Metric
NAO	Network Application Orchestrator
OSS	Operations Support System
MaaS	Metal-as-a-Service
NS	Network Service
WAN	Wide Area Network
SSD	Solid-State Disk
SAS	Serial Attached SCSI (Small Computer System Interface)
MIMO	Multi-Input Multi-Output
eNB	e-NodeB
gNB	g-NodeB
USRP	Universal Software Radio Peripheral
UE	User Equipment
RAN	Radio Access Network
O-RAN	Open RAN
GPU	Graphics Processing Unit
RU	Radio Unit
VPN	Virtual Private Network
SA	Stand Alone
MANO	Management and Orchestration
VNF	Virtual Network Function
OSM	Open-Source MANO
AiO	All-in-One
DMZ	Demilitarized Zone
CPE	Customer Premises Equipment
NUC	Next Unit of Computing
VRF	Virtual routing and Forwarding
NFV	Network Functions Virtualization
NFVCL	NFV Convergence Layer
MetalCL	Metal Convergence Layer
LLDP	Link Layer Discovery Protocol

PNF	Physical Network Function
VLAN	Virtual Local Area Network
IaaS	Infrastructure-as-a-Service
PaaS	Platform-as-a-Service
NFVO	Network Functions Virtualization Orchestrator
KNF	Kubernetes-based Network Function
SBA	Service Based Architecture
SMF	Session Management Function
UPF	User Plane Function
API	Application Program Interface
UC	Use Case
SIM	Subscriber Identity Module
ExFa	Experimentation Facility
nApp	Network Application

Executive Summary

The purpose of the deliverable is to report the description of the infrastructure, including the DevOps testbed and the ExFas, along with the procedures followed and the results obtained for the system-level validation activities.

It includes a detailed description of the DevOps testbed's structure and capabilities. This infrastructure has been used to derive Core Validation Metrics (CVMs) regarding the verification of NAO-OSS interworking, as well as the evaluation of the 5G-INDUCE Platform performance with regard to the deployment of nApps. In particular, the CVM regarding interworking of the NAO and OSS components of the 5G-INDUCE platform (time elapsed from the triggering of the creation of a slice through the NAO slice intent module to the OSS until the completion of nApp deployment by NAO, once the slice is fully established and operational), has been evaluated on both the DevOps testbed at CNIT premises and the 5TONIC/Ford/UPV ExFa that spans over the 5TONIC project's, Ford's and UPV's premises in Madrid and Valencia, Spain, in order to compare deployment operations over two significant alternative facilities.

The DevOps testbed is also employed to perform the onboarding and deployment validation of the nApps regarding the various use cases, to obtain a comparison over a uniform environment. Finally, the readiness of the ExFas and their ability to host the integrated 5G-INDUCE solution has been tested in the three specific environments in Spain, Greece and Italy.

1 Introduction

1.1 Deliverable Purpose

The deliverable has three basic purposes:

- To provide a detailed description of the DevOps testbed and the processes followed for the nApp use case adaptation and verification.
- To provide the validation and the outcomes of nApps' performance evaluation regarding the NAO-OSS interworking over the DevOps testbed and a specific infrastructure in ExFa Spain,
- To evaluate the readiness of the ExFas and their ability to host the integrated 5G-INDUCE solution, with respect to the various use cases.

1.2 Relation with other Deliverables and Tasks

The present Deliverable has relations with D5.1, for what regards the description of Core Validation Metrics (CVMs) and of the methodology for their collection. The measurement tests that appear here concern the onboarding and deployment of Use Cases' nApps on the platform and the readiness of the ExFas. Specific validation and measurements per Use Case will be reported in D6.2.

1.3 Deliverable Structure

The document is organized as follows. Section 2 provides a detailed description of the CNIT DevOps testbed, concerning its components, the arrangement of the physical infrastructure, and its management. Section 3 reports a summary of the taxonomy proposed in D5.1 for the validation of the outcomes of 5G-INDUCE. Section 4 evaluates the readiness and the operativity of the 5G-INDUCE platform, including its own deployment, and reports the CVMs of interest obtained on the DevOps testbed and on the 5TONIC/Ford/UPV of ExFa Spain. The DevOps testbed has been used also for the onboarding and deployment validation of the nApps that can be found in Section 5, since the usage of the DevOps for this testing campaign allows providing the same environment to all the nApps in the absence of the specificities of each ExFa, which will be investigated in the actual performance evaluation in D6.2. Section 6 investigates the readiness of the ExFas and their ability to host the integrated 5G-INDUCE solution. Section 7 contains the conclusions.

2 The DevOps Testbed

The CNIT S2N testbed located in Genoa (IT) is a multi-layered hardware and software facility for the advanced experimentation and demonstration of 5/6G, Edge and Cloud Computing technologies. It is specifically conceived to host multiple isolated tenant spaces, or "islands" (e.g., project environments) that can emulate complete 5/6G network environments, as well as to manage and configure their respective physical/virtual resources through a Metal-as-a-Service (MaaS) approach and the software elements through Red Hat Ansible. **The testbed is part of the Scientific Large-Scale Infrastructure for Computing/Communication Experimental Studies (SLICES) [1] project, which has been selected to be part of the 2021 roadmap of the European Strategy Forum on Research Infrastructures (ESFRI) [2]**.

In a nutshell, the testbed has hardware capabilities for setting up 5/6G network environments, complete with User Equipment (UE), UE emulator, radio/wired access, programmable connectivity, security system and (distributed) computing domains, as well as equipment useful for performance evaluations such as traffic generators and power monitors. Based on these infrastructure resources, the testbed platform integrates both proprietary and open-source software to realize and dynamically manage the islands on top – specifically, the involved base 5/6G and application-/slice-specific network services (NSs), (distributed) computing domains and WAN overlay networks.

The testbed infrastructure, shown in Figure 1, is an integration of both general- and special-purpose equipment to realize underlying 5/6G networks substrates and relative slices, as well as edge-cloud computing resources.

At the infrastructure level, the testbed currently hosts the following equipment:

- 35 servers (1300 cores, 10 TB RAM, local & central SSDs/SAS storage >100TB),
- 8 high-speed switches (918 ports from 1GbE to 100GbE),

Figure 1: The CNIT S2N testbed.

- Base stations: 2x Amarisoft Callbox 5G gNB MIMO 4x4, + 3x LTE+ eNBs (USRP-based), 1x O-RAN sub-6GHz RU, 1x O-RAN mmWave RU, 2x 1x O-RAN sub-6GHz AiO femtocells
- 1x Amarisoft 5G UE Emulator
- 4x GPU Nvidia A100
- 1x P4 (Tofino-enabled) switch
- Power monitors, hardware traffic generators, hardware firewalls, 12 UEs (drones, tablets, modems, etc.).
- The test bed is flexibly managed through a Metal-as-a-Service approach (OpenStack and Kubernetes instances as-a-Service)
- Site-to-site and client-server VPNs.
- Testbed-wide time synchronization through IEEE-1588v2 PTP Grandmaster Clock (approx. 20 ns precision)

Thanks to these assets, the testbed can currently deliver a complete 5G SA solution and research activities are in place to evolve beyond 5G. Furthermore, the hardware assets are complemented with a set of software products that allow for the deployment and management of vertical applications and related slices according to IaaS and PaaS paradigms and to even create and handle wide-area networks: in fact, not only different VIMs and Kubernetes clusters can be created, but it is even possible to view the available computing resources as edge or cloud thanks to the presence of a delay generator. The main software artifacts are the following:

- OpenStack, Open-Source MANO (OSM), Kubernetes, etc.
- CNIT S2N/Unige TNTLab OSS
- 4G/5G and networking VNFs
- DPDK-based delay and packet loss emulation
- ONOS SDN Controller (P4 and OpenFlow)
- Vertical Application Orchestrator
- Wide Area Infrastructure Manager
- Observability Stack (Prometheus, ELK, etc.)
- Keysight IxCore
- Automated and personal VPNs.

2.1 The Physical Infrastructure

Figure 2 shows the hardware resources available in the testbed. The stack of servers on the bottom right represents the central nervous system of the infrastructure and is the only persistent installation of the testbed. In fact, the other resources, both physical and logical, are delivered to the experimenters on-demand, in an as-a-Service fashion. The controller performs resources' assignment and network configurations, while the OpenStack 0 VIM (OS0 hereinafter) offers several services that are common to all experiments and are delivered by the management switch (dark grey left to the Controller in Figure 2) through an out of band management network. OS0 also hosts the demilitarized zone (DMZ). The actual operations for creating the isolated islands for the experimenters are described in the next section.

On the left of Figure 2 are depicted two stacks of servers that correspond to the racks in Figure 1 and represent the programmable computing resources of the testbed. Along with the computing resources, a number of OpenFlow switches and one P4 switch [3] are made available for the experimenters, as well as access and user devices. Access devices include IEEE 802.11v6 access points, 5G CPE routers [4], O-RAN radio

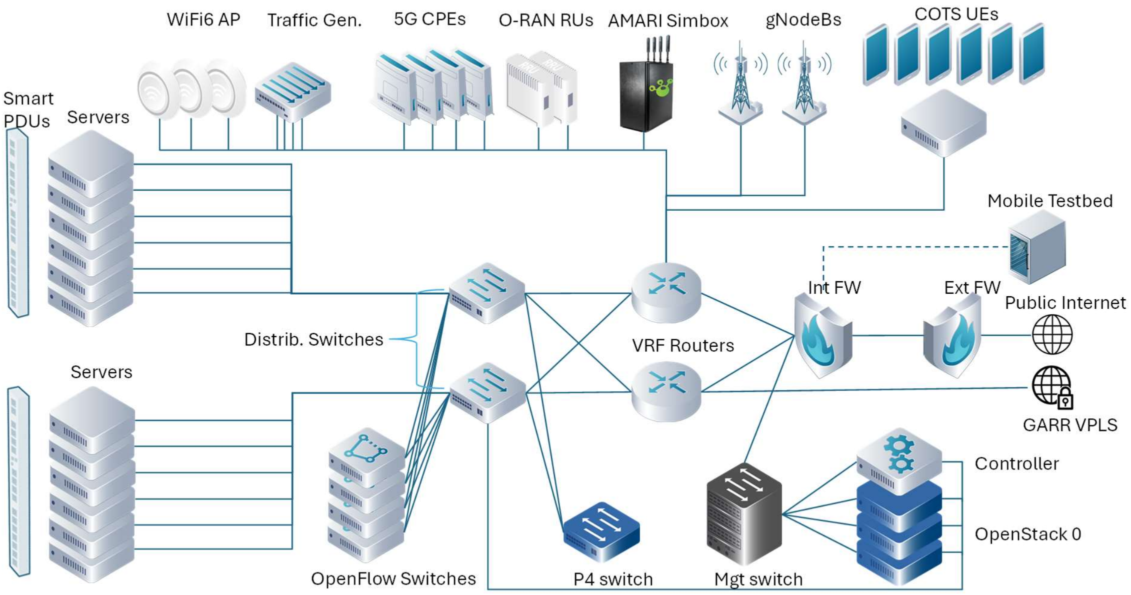


Figure 2: The physical infrastructure of the S2N testbed.

units, as well as two physical gNodeBs. Regarding UEs, we have an Amarisoft UE simulator [5] and several 5G smartphones. The latter are made available remotely thanks to a NUC that hosts Guacamole [6] and screencpy [7] to mirror the screens via HTML. The set of resources made available for the experimenters is completed by a traffic generator and two PDUs.

The physical topology of the testbed is realized through a network of switches with speeds spanning from 10 to 100 Gbps. The connectivity of the islands is handled through Virtual Routing and Forwarding (VRF) dedicated to each island, whose isolation is made possible by the internal firewall: all traffic from an island is forwarded to this firewall that prevents its exchange with the other islands via specific rules. The external firewall instead is used to manage the connectivity towards the public Internet. In addition, a direct link to the National Research and Education Network (GARR) [8] will be established shortly to interconnect the testbed with the other facilities involved with the SLICES Infrastructure.

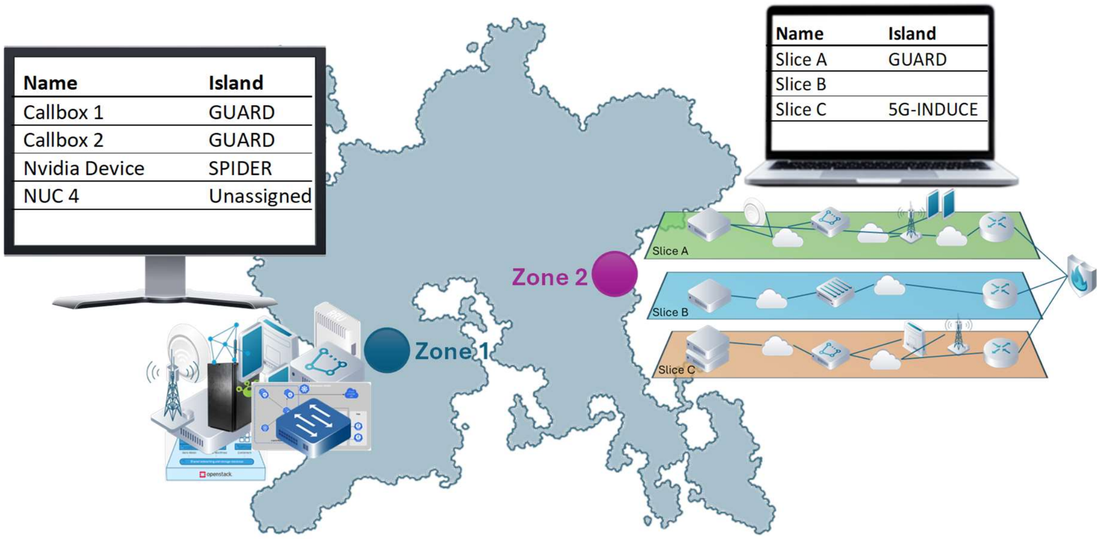
The entire physical infrastructure is monitored by means of an observability framework that exploits Simple Network Management Protocol (SNMP) interfaces and Prometheus exporters, collects the results into time series and exposes them to the experimenters through a database made available on the OS0 DMZ.

2.2 Management of the Infrastructure

In order to guarantee all the experimenters on our testbed with dedicated, customizable resources to fit their needs, a meta-orchestrator has been designed to provide them with network overlays of interconnected devices that can be selected, set up, reconfigured and released in an as-a-Service fashion thanks to several tools that have been created and refined within the research activities that have been carried out in the S2N Lab for over a decade. The meta-orchestration solution is composed of two main components, namely the Metal Convergence Layer (MetalCL) and the NFV Convergence Layer (NFVCL) [9] that, in Layman's terms, handle the creation of the islands and their lifecycle management, respectively. The current solution provides the best balance between automation and customization while guaranteeing a confined environment on which tests can be run securely and without the risk of being influenced by other experiments carried out in the same premises at the same time.

When the request for an island is received, the Controller selects the most suitable servers to be devoted to that island and asks the MetalCL, a software artifact initially developed by CNIT within the SPIDER Project, to proceed with their setup. First, Canonical MaaS [10], integrated in the MetalCL, is used for the discovery, commissioning, deployment, and dynamic configuration of the physical servers, and uses REST APIs to initiate bare-metal level changes, installing operating systems as needed. Then, an Ansible engine [11] performs software installations, deploys applications, and reconfigures the operating systems across the bare-metal servers provisioned by MaaS. During this phase there is the creation of the Kubernetes clusters and OpenStack projects on the islands. Finally, an additional Python service called NetCL performs automated discovery, employing the LLDP protocol to uncover the physical topology and configure overlay networks to facilitate the seamless hosting of OpenStack and Kubernetes, and also actively manage interconnectivity among servers. The NetCL action is crucial to provide interconnectivity while guaranteeing isolation. In short, VLANs are used inside the islands for the connectivity of clusters and projects and a VRF dedicated to the island, that has said VLANs as interfaces, is created in one of the distribution routers shown in Figure 2. Rules on the two firewalls, as explained in the previous section, prevent traffic exchange among islands while providing them with connectivity to the Internet and allowing access from remote experimenters via VPNs.

VRFs are also used for assigning physical devices (be them UEs, OF switches, etc., they are all seen as Physical Network Functions - PNF) to islands: each PNF is connected via VLAN to a VRF that, when connected to the VRF of an island, gets to share its routing table and as such is reachable and usable by the experimenters.



Name	Island
Callbox 1	GUARD
Callbox 2	GUARD
Nvidia Device	SPIDER
NUC 4	Unassigned

Name	Island
Slice A	GUARD
Slice B	
Slice C	5G-INDUCE

Figure 3: Example of two zones over a geographical area with different programmability levels, Zone 1 exposing programmability at MaaS level and Zone 2 exposing a catalogue of slices.

The capabilities of the MetalCL extend beyond the borders of our testbed facility: in fact, it has the ability to manage several federated infrastructures by conceptualizing the various facilities as “zones”, segregated collections of hardware and software resources, each tailored to offer specific functionalities. MetalCL allows to handle the peculiarities of the individual zones, such as the available hardware and the level of programmability that they expose to the users; an example can be seen in Figure 3. Indeed, each zone may provide a diverse spectrum of resources that include servers dedicated to the bare metal deployment of OpenStack and Kubernetes, alongside instances of both platforms, and can be summarized as follows:

- *No programmable domains*: a catalogue of the pre-allocated network slices and edge processes and of their configuration parameters is made available to the experimenters.
- *Configurable Domains*: programmability is handled by the infrastructure through a black-box system that evaluates the feasibility of the requested slice update.
- *Domains programmable at PaaS virtualization level*: This type of domains corresponds to environment where a full Kubernetes cluster is provided as a programmable resource, so lifecycle management operations are allowed only on cloud-native services.
- *Domains programmable at IaaS virtualization level*: Like the previous case, only IaaS (and PaaS) network services can be built and managed.
- *Domains programmable at hypervisor Operating System level*: In addition to the capabilities foreseen in the previous case, this will allow to install and fully customize IaaS and PaaS environments (e.g., installing special versions of OpenStack and Kubernetes, selecting additional packages, etc.).
- *Domains programmable at MaaS level*: This case represents the maximum possible programmability level and is what is offered by our testbed: commission and configure bare-metal resources (hardware servers and hardware network devices like switches, routers, firewalls) to dynamically create, scale, and terraform IaaS and PaaS environments.

Once the MetalCL has finished configuring an island, a dedicated project is created on OS0 to host the management software dedicated to that island. As shown in Figure 3, which presents an example of a deployed island, OS0 has also a persistent part devoted to software that can be shared among islands, such as vertical orchestrators, OSSs, metric collectors (e.g., Prometheus [12], ELK [13]), etc. Instead, the dedicated project contains pieces of software that are specific for managing the granted programmability level. The NFVCL is particularly noteworthy for providing a level of abstraction for the flexible and high-level management of the complete lifecycle orchestration of network services, VNFs and PNFs instantiated in the 5G infrastructure.

More precisely, the NFVCL can build and dynamically manage complete network environments (e.g., a 5G core network with different NG-RANs) by composing and orchestrating through an NFV Orchestrator (usually the ETSI Open-Source MANO) multiple NFV services in a joint and coherent fashion. This logical group of NFV services realizing a certain network environment is managed within the NFVCL by a “blueprint”, a deployment template to support ad-hoc operations for specific implementations of high-level network ecosystem function types, like a 5G system, a network security toolchain, etc. The NFVCL was designed to support multiple types of blueprints, and multiple instances per type. The number and the type of NFV services (e.g., the number and the specific implementation of NG-RAN NFV services in a 5G network) to be used is dynamically selected by the blueprint according to the request parameters (e.g., the geographical coverage requested for the slice).

During Day-0 operation, the NFVCL produces and onboards the ETSI SOL006 [14] descriptors of services and of related PNFs, VNFs, KNFs onto the NFVO by defining the needed number of virtual links and of virtual resources to be applied. In Day-1 operations, the NFVCL requests the NFVO to instantiate network services by selecting the computing facilities and networks where to attach network functions. At Day-2, the NFVCL produces the configuration files and commands for each of the deployed VNFs and applies them through the VNF Managers at the NFVO. Moreover, the NFVCL has the ability to create and fully manage Kubernetes clusters over IaaS environments. Therefore, where IaaS programmability is possible, the NFVCL can create the cluster and manage container-based VNFs over it. The whole procedure involving all the steps performed from the MetalCL and the NFVCL is reported in Figure 4

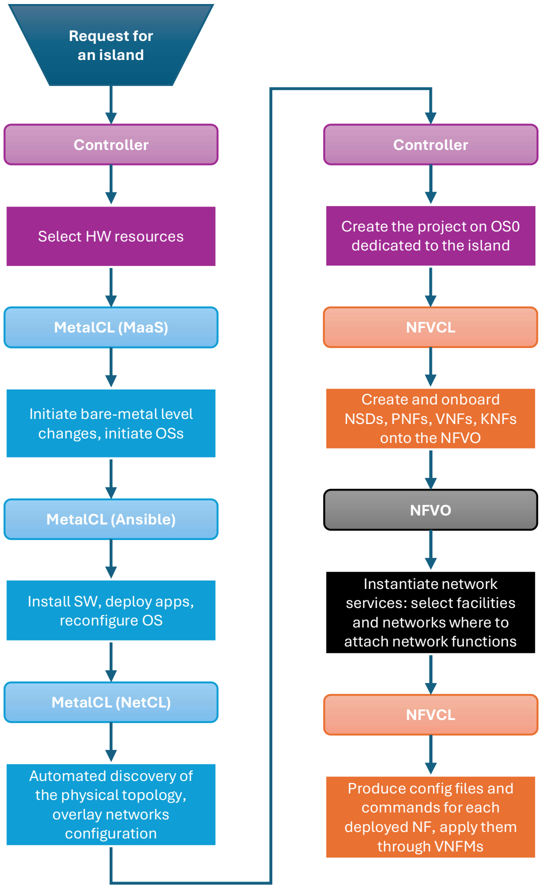

Request for an island
Controller
Select HW resources
MetalCL (MaaS)
Initiate bare-metal level changes, initiate OSs
MetalCL (Ansible)
Install SW, deploy apps, reconfigure OS
MetalCL (NetCL)
Automated discovery of the physical topology, overlay networks configuration
Controller
Create the project on OS0 dedicated to the island
NFVCL
Create and onboard NSDs, PNFs, VNFs, KNFs onto the NFVO
NFVO
Instantiate network services: select facilities and networks where to attach network functions
NFVCL
Produce config files and commands for each deployed NF, apply them through VNFMs

Figure 4. Flow diagram representing the steps performed by the MetalCL and the NFVCL to setup the infrastructure component and networking of an island in the S2N testbed.

Moreover, in the example of Figure 5, we can notice the VRF router and the three networks that have been created inside this representative island: the "Control Network" is found by default in every island as it is used for the interconnection with the dedicated project on OS0, while the other two are specific for this example, "Mgmt Network" being used for interfacing VNFs, KNFs and PNFs for management purposes and "WAN" being used by OpenStack to interconnect several projects.

Inside the OpenStack n project, we can see a 5G core deployment that is expanded in Figure 6. The base station can be a physical gNodeB or an emulated one, be it an O-RAN system, an Amarisoft simulator, etc., regardless it is bound to the island via the VRF as shown in Figure 5. UPFs can be shared among the clusters and projects inside the island and in turn attached to vApps handled by the NAO instance in the persistent part of OS0. Regarding the SBA, the number of instances of a NF and their cluster can be fully customized according to the demands of the experimenter: for example, one could decide to replicate an SMF, or to instantiate the NFs on one or more clusters. It is even possible to create new, ad-hoc NFs from scratch (BYONF – Bring Your Own NF in Figure 6) and interface them with the instantiated core.

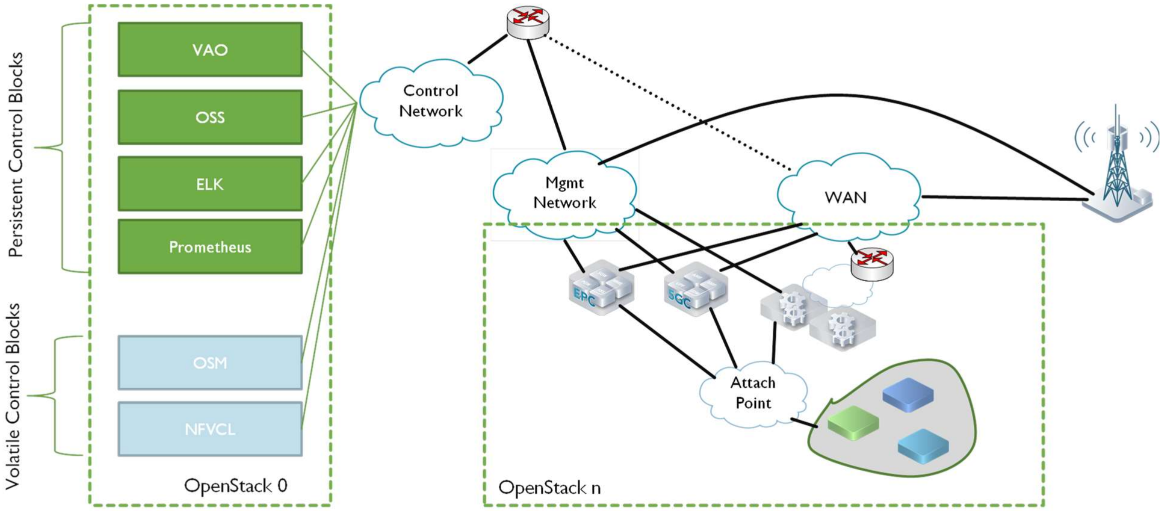


Figure 5: Example of an island architecture and interfaces

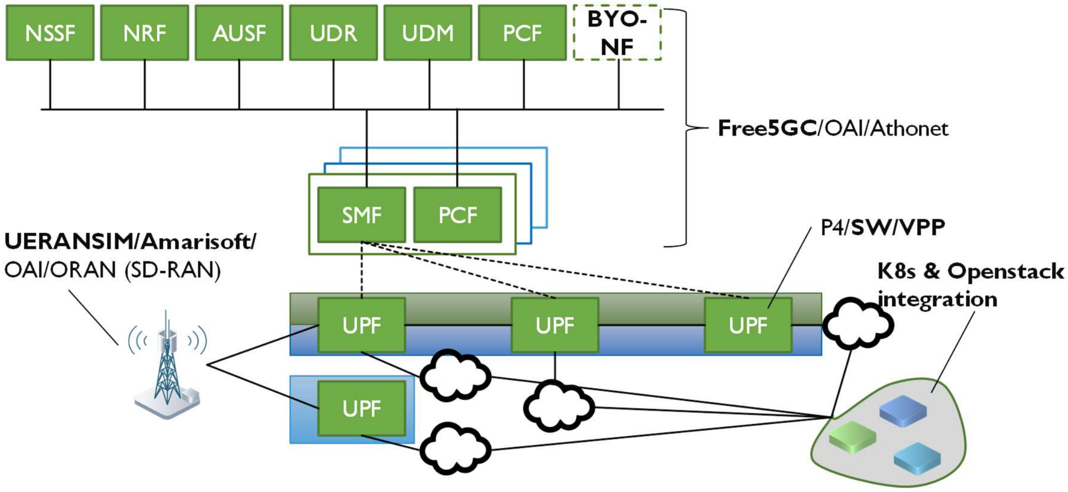


Figure 6: Deployment of a 5G core in the example island.

3 System-level Validation Activities

This section reports a summary of the taxonomy proposed in D5.1 for the validation of the outcomes of 5G-INDUCE. The proposed methodology distinguishes between system-level and service-level validation activities: the latter refer to the assessment of the solution as a whole, with the validation of the nApps in the project ExFas and is the subject of D6.2. The system level validation, instead, covers the preparation, the testing and the collection of the results for the individual components of 5G-INDUCE.

D5.1 defined a set of Core Validation Metrics (CVMs), which includes a description of the metrics and of the methodology for their collection. The list of CVMs, reported in Table 1 in D5.1, is used for all the systems under test; however, for some CVMs a different measurement methodology may be adopted in case an nApp is being tested instead of an architectural component. Moreover, some CVMs may not be applicable for each system under test and only the ones of interest, which have been originally identified in D5.1 as well, are reported in the following.

The remaining of the document covers the system-level validation activities and outcomes obtained at the end of 5G-INDUCE's lifetime. Section 4 evaluates the readiness and the operativity of the 5G-INDUCE platform, including its own deployment, and reports the CVMs of interest obtained on the DevOps testbed. The same testbed has been used also for the onboarding and deployment validation of the nApps that can be found in Section 5. It is worth noting that the usage of the DevOps for this testing campaign allows providing the same environment to all the nApps in the absence of the specificities of each ExFa, which will be investigated in the actual performance evaluation in D6.2. Finally, Section 6 investigates the readiness of the ExFas and their ability to host the integrated 5G-INDUCE solution.

4 5G-INDUCE Platform Readiness and Performance Validation

Tests focus on the verification of NAO-OSS interworking and on the evaluation of the 5G-INDUCE Platform performance with regard to the deployment of nApps.

Description

Validation of the 5G-INDUCE platform readiness, including the interplay between the NAO and the OSS, the management of nApp deployment and high-level orchestration of network and compute resources.

Purpose: To ensure that the exchange of intents with the NAO and the assignment/modification/release of resources are functioning properly, allowing the platform to proceed with nApp deployments over its interfaced infrastructure.

Network Deployment

Tests were performed over the DevOps testbed at the CNIT premises in Genoa, Italy, as well as over the 5TONIC/Ford/UPV ExFa that spans over the 5TONIC project's, Ford's and UPV's premises in Madrid and Valencia, Spain.

The description of the solution for the integration of the 5G-INDUCE platform with the aforementioned ExFas is available in the deliverable D3.5.

Running the tests over both the Spanish and the DevOps ExFas has allowed to assess NAO and OSS interworking and performances in two meaningful scenarios; in fact, with reference to the ExFa classification defined in D3.4:

- the Spanish ExFa corresponds to a "configurable ExFa", in that it embeds
    - a 5G Core controller that exports towards the 5G-INDUCE Platform a set of APIs that enable the configuration, by the OSS, of a subset of parameters on a pre-deployed network slice,

      and
    - a pre-deployed Kubernetes cluster, where a set of namespaces are accessible by the NAO;
- the DevOps ExFa falls under the "IaaS-level programmable" category, in that the OSS is able to deploy 5G slices by orchestrating related VNFs and fully configuring PNFs, as well as to create Kubernetes namespaces and grant access to the NAO.

Testing Procedure

Step 1: Bootstrap the 5G system.

Step 2: Verify the proper exchange of messages between NAO and OSS.

Step 3: Given the requests in the slice intent, check the availability of the required computing resources and NFs.

Step 4: Provide the NAO with a tentative slice. If accepted, proceed with the next steps; otherwise, re-negotiate a new slice intent.

Step 5: Create the tenant spaces for the NAO and for the NFVO. Testing of NAO connectivity to the cluster.

Step 6: Once successful, then proceed with the deployment of the nApps and NFs.

Step 7: Verify the nApp PODs are running.

CVM-3: total time elapsed from the triggering of the creation of a slice through the NAO slice intent module to the OSS until the completion of nApp deployment by NAO, once the slice is fully established and operational.

KPI values in the range 31 – 99 [s]

Remarks

Table 1 lists the measurements performed on the DevOps and Spanish (FORD/5tonic/UPV) ExFas, related to the deployment of a meaningful subset of Use Cases, supported by different slice types. Table 1 reports two values:

- the time the OSS needed to elaborate the "slice intent" received from the NAO and either deploy or configure the 5G slice and Kubernetes namespaces; the elapsed time is measured from the moment when the slice-intent message is issued (by the NAO) until the moment when the relevant slice-reply is received (by the NAO itself);
- the time the NAO needed to deploy the nApp containers once it receives the confirmation of the availability of the slice from the OSS; the elapsed time is measured from the moment when the deployment command is issued until the moment when all relevant containers are up and running.

OSS processing time largely varies depending on the operations the OSS is requested to implement on the ExFa infrastructure: it is minimal in a case such as the Spanish ExFa, where OSS only sets a few parameters of a pre-deployed slice; it is much longer when the OSS needs to fully deploy a slice by interworking with the DevOps 5G Core (which corresponds to 5GC, in case of the listed tests). In the latter case, the OSS takes most of the time needed to complete the sequence from slice negotiation (between NAO and OSS) to nApp deployment and start-up.

	Testbed	*Use Case*	*Number of PODs*	*Slice type*	*Measured Times*		
					OSS operation – Slice setup	**NAO operation - nApps Deployment**	**Total time**
1	Ford/5TONIC/UPV	UC #1	3	eMBB	8 s	25 s	33 s
2	Ford/5TONIC/UPV	UC #2	2	eMBB	8 s	16 s	24 s
3	Ford/5TONIC/UPV	UC #3	4	eMBB	8 s	23 s	31 s
4	DevOps	UC #5	4	eMBB	86 s	13 s	99 s
5	DevOps	UC #5	4	URLLC	82 s	15 s	97 s
6	DevOps	UC #6	1	eMBB	90 s	4 s	94 s

Table 1 – Platform Performance Validation - Test results

5 Network Application Onboarding and Deployment Validation

The developers that need to construct their network application need to prioritize both functionality and the confidentiality of their components. Each network application element is designed to operate autonomously, exposing relevant information and accepting necessary inputs through various communication frameworks like REST API, message bus, and pub-sub. In 5G-INDUCE, containers facilitate this process, allowing developers to use their preferred or most familiar programming languages to create applications or network functions. Inside the 5G-INDUCE project the network application onboarding and deployment validation methodology is a meticulously structured process designed to ensure the integrity and functionality of network applications.

Once a functionality is developed, it must go through a stringent process of compilation, building, and packaging in compliance with 5G-INDUCE standards (see D4.3, section 3.3). This involves providing a Dockerfile or Helm chart that specifies the steps for these processes. By doing so, multiple application or network functions can be seamlessly linked to form a comprehensive network application, with each function regarded as a distinct component. A critical step in this methodology is the pre-onboarding process, where developers must submit the code and/or binaries for each nApp component along with its test code. This step is mandatory and ensures that every software artifact stored and prepared for deployment meets the specific requirements and deployment criteria of the 5G-INDUCE platform. This rigorous validation process is essential to verify that each component functions correctly and securely before the onboarding to the platform happens.

The onboarding and deployments have been performed in the DevOps testbed in the premises of CNIT.

5.1 Methodology

The onboarding methodology for the 5G-INDUCE platform is designed for universal applicability across diverse use cases. This standardized approach ensures seamless integration and management of network application components through the NAO. Key aspects of this methodology include the registration of components and deployment commands, all thoroughly described in Deliverables 4.1 and 4.3.

5.1.1 nApp preparation steps for deployment

The following steps are defined for any type of nApp that is to be deployed through the 5G-INDUCE platform:

1. Containerization of the services to be deployed.

Convert the services into containers, encapsulating the software along with its dependencies to ensure consistency across different environments.

2. Store and validate the nApp component code/images to the repository.

Upload the container images and/or component's code to the 5G-INDUCE repository for centralized management. The developer must also provide a descriptor like Docker-Compose file or a Helm chart that includes the deployment parameters necessary for validating the application.

This repository ensures secure storage and availability of images for deployment while maintaining version control. Automated pipelines for compiling, testing, packaging, and building the code into a binary file can be used. This automation along with the tests guarantees that components are consistent before the deployment starts.

3. Registration and composition of an Application.

After validating the code/images, the registration of the application components and the composition of network application graph starts within the NAO. Each application may include multiple components that execute parts of the application logic, structured in a directed acyclic graph. For each component there must be declared the minimum execution requirements, a heartbeat – aka, a way for the 5G-INDUCE Platform to verify that the component is running or not – environmental variables and more (see D2.1, section 3.2). After registering individual components, the developer can compose and register the application graph via the NAO interface. This involves linking components to create a coherent application structure, with users able to search for components and visually connect them.

4. Declaration of deployment requirements.

In the post composition phase, users declare resources and high-level network requirements that must be fulfilled by the OSS. These constraints ensure the application placement meets specific network functionalities and this is where the application is turned into network application, i.e. the application is deployable over the 5G-INDUCE platform and includes the required networking features. The information related to the selection of the deployment location (according to the registered virtual infrastructure managers (see D3.5, section 4.3), as well as the information about application-specific requirements, is formatted and sent to the OSS, which allocates resources and manages network slices (see D3.5, section 4) .

5. Application Deployment.

Following the completion of the previous steps, the application is deployed. Feedback is provided through a visual representation of the application graph's status, with runtime logs accessible for monitoring deployment status and basic metrics for each component.

The validation process primarily measures the time required to deploy the application on the desired programmable resources. It provides detailed timing information regarding the orchestration of the application, ensuring that the deployment process meets expected performance standards and operational efficiency.

5.1.2 Network Application Onboarding-Registration-Deployment timings

The core validation metering for network applications relates to the time required from when the end-user registers the application until the application is operational. This is further split into two timings, one related to the application registration and the composition of the network application graph (pre-deployment time) and a second one related to the actual deployment through the platform.

Application registration and composition timings:

- Component registration: The process of registering application components within the NAO is timed.
- Resource declaration: The time required for users to declare resources and high-level network requirements in the NAO is measured. This ensures the application meets specific network functionalities.
- Graph composition: The time taken to compose the network application graph, linking multiple components in a directed acyclic graph, is recorded. This step includes declaring execution requirements, heartbeat mechanisms, and environmental variables.

Deployment execution and post-deployment validation timing includes:

- Resource Allocation: The OSS allocates resources and manages network slices based on the declared requirements. The time taken for resource allocation is measured from the moment the slice intent request leaves the NAO until the OSS replies back with the realized slice in the NAO (see Table 1).
- Application Deployment: The overall deployment time, from initiation to completion, is recorded. This includes monitoring the visual representation of the application graph’s status and accessing runtime logs for each component
- Initial Run Verification: The time required to verify that all components are running correctly post-deployment is measured.

The following dependencies have been identified and taken into consideration for the validation of the various use cases (see next sub-section 5.2.1).

a. **Number of containers**: The number of containers does not directly impact the overall application registration and composition timings, as all containers are registered in parallel. However, the configuration of each individual container affects the application registration timing.

b. **Depth of Dependency:** This term refers to the number of containers upon which a particular container relies. It defines the extent of the container's dependency chain, indicating how many other containers must be in place and functioning correctly for the dependent container to operate as intended.

- For example, if container A depends on container B, and container B depends on container C, then the depth of dependency for container A is 2. This means container A has a dependency chain that includes two other containers (B and C) that must be operational for A to function correctly. If there are multiple dependency chains, the chain with the maximum dependency depth value is the one that affects the overall registration time, since these chains can be deployed in parallel.

Furthermore, it is noted that deployment time (Deployment execution and post-deployment validation column in Table 2) on the targeted system over which the deployment takes place, is as described in section 4. Here the validation is made considering the DevOps testbed (i.e., deployment over the fully programmable infrastructure).

5.2 Validation

5.2.1 Validation of Onboarding-Registration-Deployment process

Table 2 summarizes the application registration and composition time and the deployment time for each use case. Moreover, the number of containers and the container dependency depth are provided. All the timings for this validation are taken from the DevOps testbed in CNIT premises.

UC#	Number of containers	Dependency depth (maximum)	Application registration and composition (seconds)	Deployment execution and post-deployment validation [s]
1	3	2	23	Total Time: 109 Slicing Allocation Timing: 84 Application Deployment Time on Kubernetes: 25
2	2	1	17	Total Time: 104 Slicing Allocation Timing: 88 Application Deployment Time on Kubernetes: 16
3	5	1	25	Total Time: 114 Slicing Allocation Timing: 92 Application Deployment Time on Kubernetes: 22
4	3	1	17	Total Time: 105 Slicing Allocation Timing: 94

				Application Deployment Time on Kubernetes: 12
5	4	2	36	Total Time: 99 Slicing Allocation Timing: 86 Deployment Time on Kubernetes: 13
6	1	0	29 (see remark 2)	Total Time: 94 seconds Slicing Allocation Timing: 90 Deployment Time on Kubernetes: 4
7	5	2	48	Total Time: 102 seconds Slicing Allocation Timing: 81 Deployment Time on Kubernetes: 21 secs
8	6	0	9	Total Time: 99 seconds Slicing Allocation Timing: 82 Deployment Time on Kubernetes: 17

Table 2: Service Deployment Timings from the DevOps testbed

Key remarks from the validated use cases:

Remark 1: For the collection of timings (above) related to Application registration and composition across all use cases, a specialized tool had been developed to automate the onboarding process to the NAO from GitLab images and accurately record these timings. This tool, developed specifically for this purpose, automates the entire onboarding process within the NAO, ensuring precise and consistent timing measurements. By excluding potential errors or delays that might arise from manual user interactions, this tool provides reliable data, enhancing the overall efficiency and accuracy of our validation process. This tool employs parallelism to register all the necessary details to the NAO efficiently. The only factor that affects this process is the dependency (Dependency depth) between components, as a dependent component must be able to locate its non-dependent counterpart.

The tool operates in three distinct modes:

1. Silent Mode:

- Utilizes the REST API of the NAO for application registration and composition.

- Takes advantage of parallel registration (multi-threading).

2. Non-Silent Mode:

- Similar to the silent mode but performs non-parallel registration (single-threaded).

3. Video Mode:

- Records the entire registration process within the NAO for documentation or demonstration purposes.

4. Experienced GUI User:

- Involves using the graphical interface of the NAO to manually register and compose the application.

The timings for each mode vary significantly, particularly for video mode and experienced GUI user mode. For reference, registering an application with three containers takes:

- Silent mode: 0 minutes 10 seconds

- Non-silent mode: 0 minutes 42 seconds

- Video mode: 3 minutes 29 seconds

- Experienced GUI user: 3 minutes 29 seconds

The timings reported in the table above were measured using silent mode (a), ensuring the most efficient and error-free registration process.

Remark 2:

Use case 6 involves a single container that requires a large range of open ports (400 ports) to be declared. Due to the lack of NAO support for declaring a range of open ports, its port configuration was handled through a separate registry. This is the reason, the application registration and composition timing of use case 6 is high.

Remark 3: As mentioned, Deployment execution and post-deployment validation timings are including the timings that have been reported in Table 1. The time is comprised of the Slicing Allocation Time, which is reported from the slice intent / slice reply timing between the NAO and the OSS and has been analysed in Section 4 and the application deployment time, which is the time measured by the NAO from the start to the end of the deployment process on Kubernetes.

From the observations, it is evident that there is a general correlation between the number of containers, the dependency between the containers and the application deployment time on Kubernetes.

Number of Containers:

Generally, an increase in the number of containers tends to increase the Application Deployment Time on Kubernetes, likely due to the higher complexity and the need to manage more resources. For example:

- UC#6, with only 1 container, has the shortest Kubernetes deployment time of 4 seconds
- UC#8, with 6 containers, has a deployment time of 17 seconds.
- UC#7, with 5 containers, has a deployment time of 21 seconds.

However, this is not a strict rule, as other factors like dependency depth and specific configurations also play a role. Higher dependency depth can complicate the deployment process, potentially increasing the Kubernetes deployment time due to the need to manage interdependencies. For example:

- UC#1 and UC#5, both with a dependency depth of 2, have deployment times of 25 and 13 seconds, respectively.
- UC#2 and UC#4, both with a dependency depth of 1, have deployment times of 16 and 12 seconds, respectively.

- UC#3 and UC#7, with a higher number of containers and varying dependency depths, show that both factors contribute to the deployment times (22 and 21 seconds, respectively).

Both the number of containers and the dependency depth influence the Application Deployment Time on Kubernetes. Generally, more containers and higher dependency depths tend to increase the deployment time, but the specific impact can vary depending on the exact configurations and complexity of the inter-container dependencies.

5.2.2 Validation of CVM-01

CVM-01: Total time required for the end user to receive an application response message after a request message is sent by the end-user, or a triggered action.

The internal communication between components at the edge did not exceed 5 milliseconds (ms), ensuring highly efficient and prompt interactions within the system. This performance underscores the robustness of the DevOps testbed in maintaining swift internal communication, thereby enhancing the overall responsiveness and user experience. For communications requiring Internet usage, the values for specific use cases ranged from 20 to 40 ms. This performance indicates that even with the additional latency introduced by Internet communication, the system maintained a commendable level of responsiveness, thereby enhancing the overall user experience and reliability.

5.2.3 Validation of CVM-04

CVM-04: number of application bits per second transferred over a specific use case interface.

The internal bandwidth within the CNIT infrastructure is 7.50 Gbps. This bandwidth has proven to be more than adequate for all the use cases tested. Throughout the various use case implementations, no bottlenecks were identified, indicating that the network's capacity was sufficient to handle the data transfer requirements effectively without any performance degradation or delays.

5.2.4 Validation of CVM-06

CVM-06: the amount of time the end-to-end application is properly delivered according to the specified performance metrics, over the amount of time that is expected to deliver the end-to-end network application service.

For this KPI the uptime of the servers in the DevOps testbed was thoroughly monitored. Over the last 6 months, the total operational time of the system was consistently tracked. The results showed an impressive availability percentage of (60/60) x 100%, indicating that the servers maintained continuous operation without any downtime during this period. This exceptional level of uptime underscores the reliability and robustness of the DevOps testbed, ensuring that the end-to-end applications are delivered seamlessly according to the specified performance metrics.

6 ExFas Deployment Readiness

6.1 ExFa-SP

In this sub-section the validation of the infrastructure deployment readiness at the ExFa-SP prior to use case experimentation is described. Even though the infrastructure is deployed at the ExFa, it is a good practice of the experimenters to check the readiness before starting, to make sure there are no infrastructure issues, or otherwise, report to the infrastructure implementers for support.

This is done with the purpose of validating the optimal performance of the network, in order to support the identified network requirements and the validation of nApps.

Network description

The network infrastructure is comprehensively detailed in D5.2, Section 3.1. The key point to remember is that a distributed architecture has been used in ExFa-SP which involves four sites: (i) Ford's facilities, where the 5G coverage is provided with Ericsson's RAN equipment, there is a 5G UPF running that will route the user plane traffic towards and from the applications and a server for applications is also available for on-premises UC executions; (ii) UPV's facilities, where there is a server for applications acting as a Near-Edge platform; (iii) 5Tonic lab, where the Control Plane (CP) part of the 5GC is located; and (iv) CNIT's facilities, where the nApp Orchestrator is running.

Additionally, there are some configurations that have been done in the 5GC CP as compulsory pre-requisites for experimentation. Several SIMs have been provisioned in the 5GC (more specifically, in the UDM) for being used in this project. Moreover, in the SMF a Data Network Name (DNN) called "5Tonic" has been created to make the network use the UPF present at Ford's premises at Valencia. The UEs in Valencia need to have the APN "5Tonic" configured in their settings.

From the Orchestrator, the nApps can be onboarded either on the server at Ford for on-premises experimentation or on the server at UPV for Near-Edge computing experimentations. These two cases have been validated in the trials.

Testing parameters and metrics

The testing procedure was performed as defined in D5.1, section 3.2. `Ping` and `iperf` tools have been used to measure latencies and maximum data rates, while a spectrum analyzer has been used to measure signal coverage levels.

CVM-01: measurement of the end-to-end delay from the time that an ICMP packet leaves the user end device, until the reply gets back to the device.

The expected value was in the range of 10-30 ms if only one RAN segment is involved, 20-40 ms if two RAN segments are involved.

Using different pings, it was obtained that on average the latency to the Ford app server was 7-8 ms; pinging to the UPV app server resulted in average latencies of 9-10 ms and pinging to the Internet (8.8.8.8) resulted in an average latency of 20 ms. So, the expected values were met.

CVM-04: maximum data rate that can be achieved in the environment. With iperfs, the maximum achievable data rate in the channel is obtained. If the demands of the use case are lower, the 5G network will be able to handle the traffic.

The expected value was to obtain a throughput in the range of 200-250 Mbps for Downlink and 40-50 Mbps for Uplink.

Iperf3 tests were performed under a DOT (where the measured RSRP was -71 dBm) for both uplink and downlink, as shown in Figure 7.

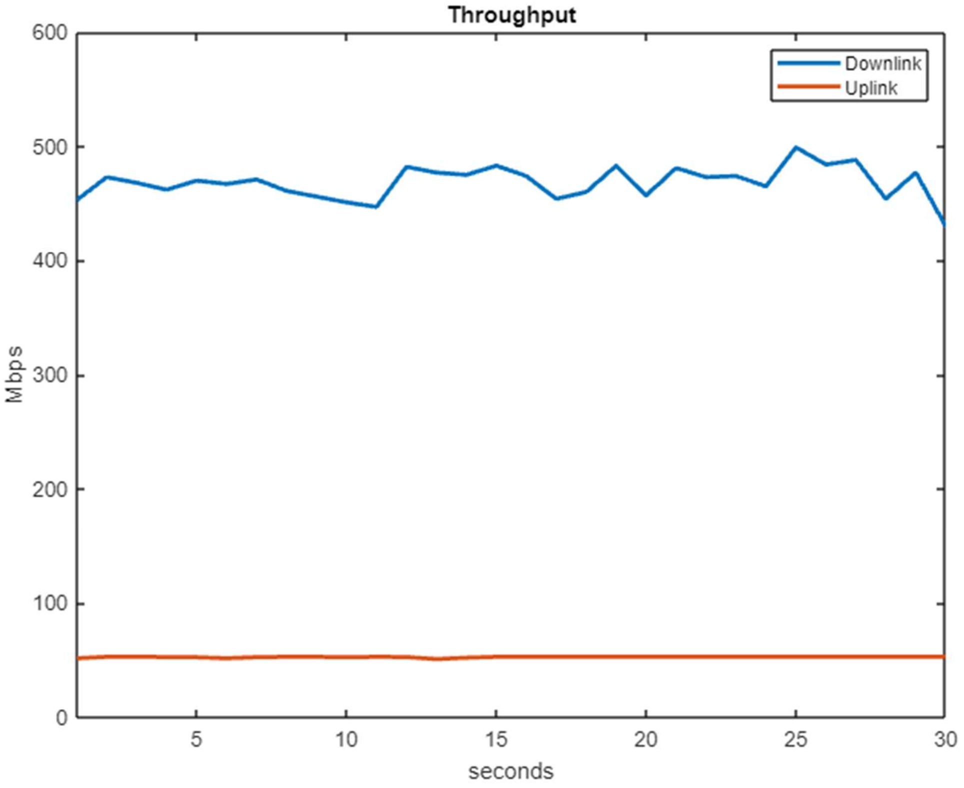


Figure 7 Uplink and Downlink throughput at the ExFa-SP

As seen in the figure above, on average the throughput obtained is 469 Mbps for downlink and 52.8 Mbps for uplink. These values are higher than the expected ones, so it can be concluded that the network provides excellent throughput values.

CVM–07/CVM-08: Network coverage and signal quality performance with respect to the specific testing area. Coverage already performed during HW installation. Mobility, signal stability and quality to be evaluated with deployed nApps.

A perfect coverage is expected within the testing area and mobility support for 20 km/h.

The RSRP along the route was measured using a spectrum analyzer, obtaining the values shown in Figure 8.

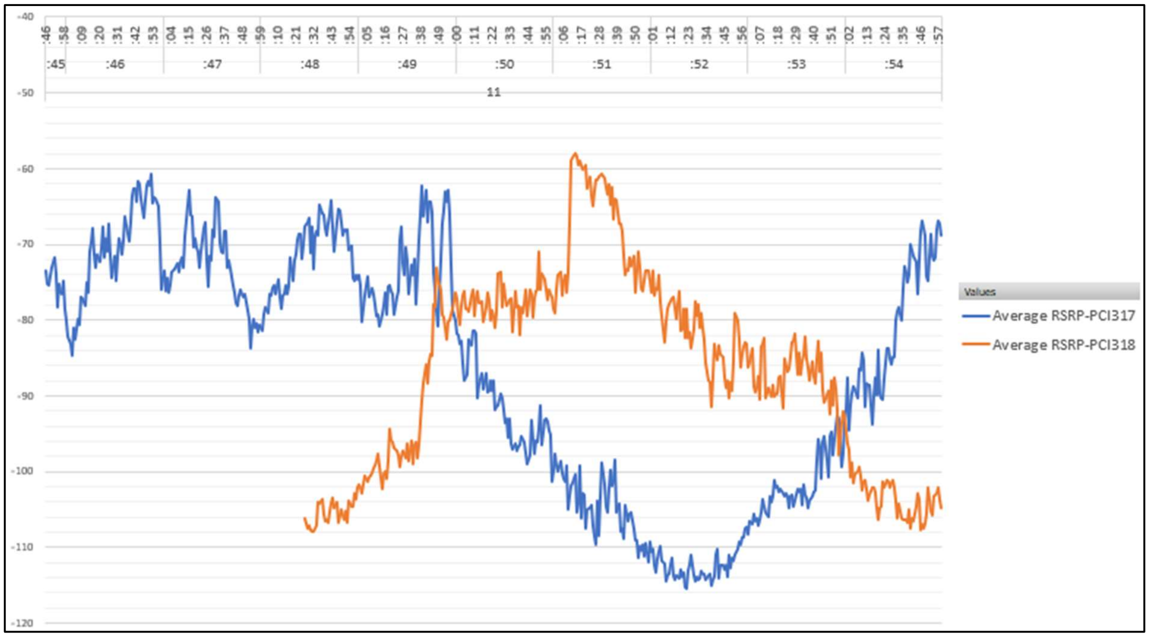


Figure 8 RSRP along the coverage area of the ExFa-SP

As shown in the figure, the RSRP stays over -80 dBm for the most part of the route, only dropping to -93 dBm in one point of the route. This means that the coverage is perfect as it was expected; the coverage along the map can be seen for both PCIs in Figure 9 (Indoor DOT coverage) and Figure 10 (Outdoor antenna coverage).

In Figure 9 the points where the handover is performed can also be seen; this handover is always performed successfully and seamlessly for the UE. The handover threshold is set to -110 dBm.

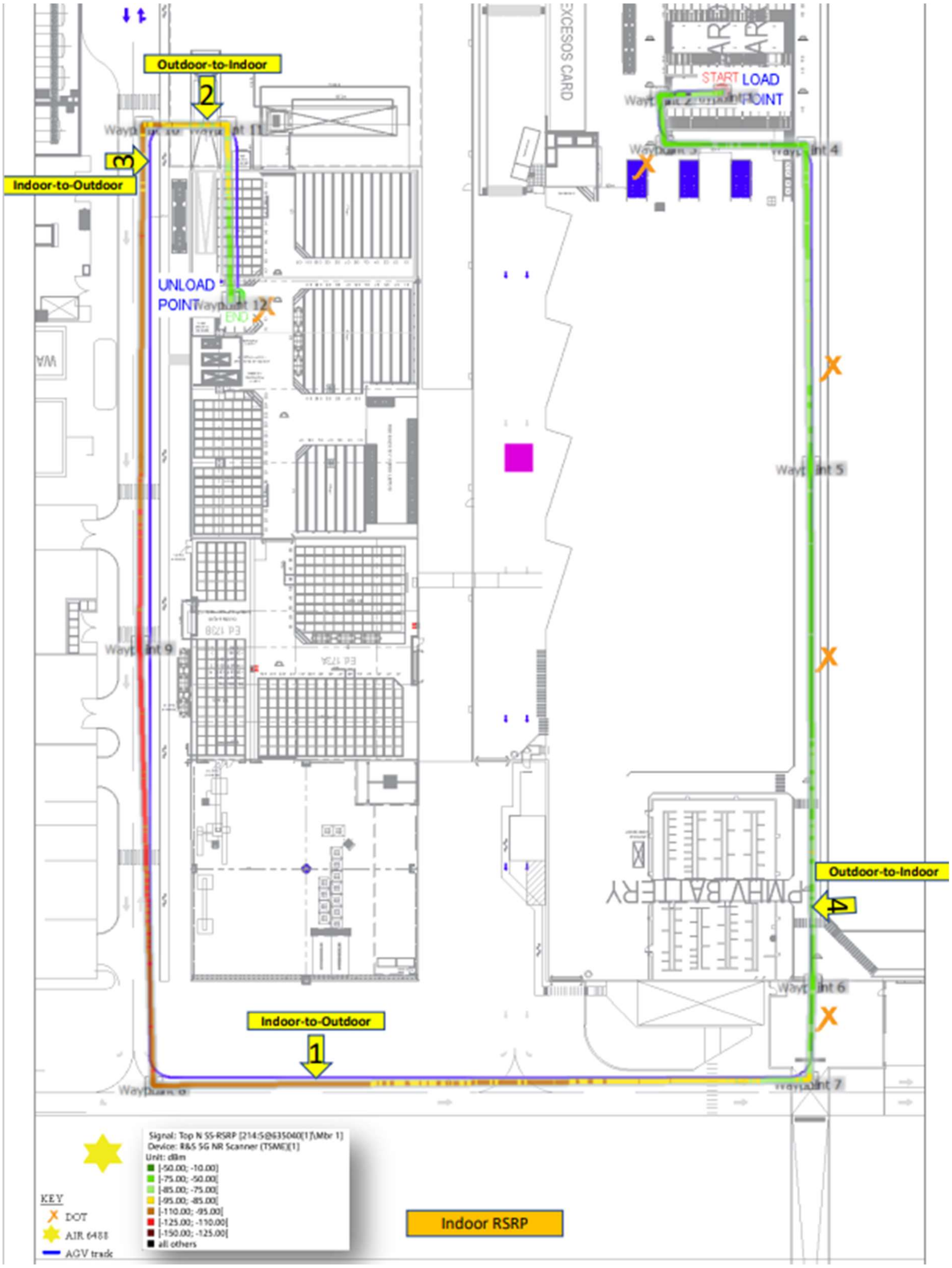


Figure 9 Handover transitions and DOT antennas coverage in ExFa-SP

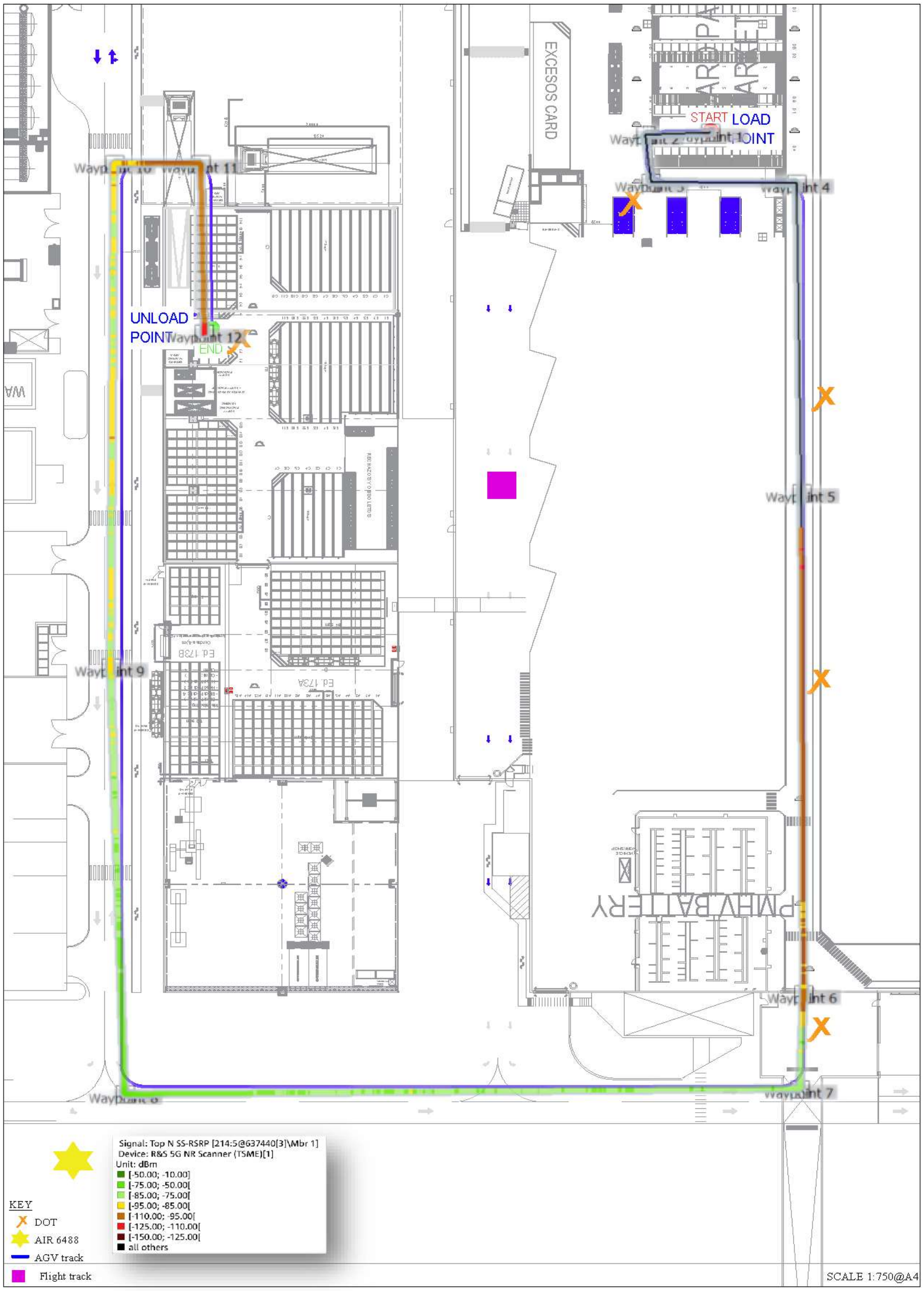


Figure 10 Outdoor antenna coverage in ExFa-SP

6.2 ExFa-GR

Description

The network test consists of measuring and validating the network required KPIs in terms of latency, throughput and network availability and reliability for the 5G testbed.

Infrastructure KPIs measurements will start from the end device site and end at packet core site. KPIs measurements will be performed for each type of service (service-slice aware).

Purpose: To validate the optimal performance of the network, in order to support the identified network requirements and the validation of nApps.

Network Deployment

The ExFa-GR was designed to host UC4, UC5 and UC6. Aiming to support the requirements of these UCs, an E2E 5G SA network was installed. Four indoor antennas, one outdoor antenna, an IRU and a BBU were installed in the demo site in PPC premises, while the packet core and an extra server hosting the nApps instances were in OTE premises. The two sites were connected using the POTP network.

To test network performance, end-to-end measurements were made using IPerf client-server interface. An IPerf server was installed in the network, on the packet core side, while the IPerf client was installed on the UE. Using this setup, throughput, latency and packet loss were measured. The packet loss results were used to draw conclusions about the availability and reliability of the network.

The UEs used were mainly 5G-gateways connected to devices such as drones, laptops and tablets (Figure 11), depending on the needs of each UC. More specifically, a Nokia FastMile 5G Gateway and a Teltonika TRB500 were used in ExFa-GR as UE.

Further details on the infrastructure are provided in D5.2.

Testing Procedure

- Initially, the IPerf client was installed on the device that would be connected to the network through the 5G-gateway.
- The 5G-gateway was registered in the network using one of the available slices. More specifically, a default slice was defined, which was used for the first registration of the UE, and then, within the framework of the implemented slice mechanism, there was the possibility to change this slice. It is worth noting at this point that the way slices work is based on the "minimum guaranteed resources" principle. Based on that, as long as there are available resources, the UE can enjoy the maximum capabilities of the network.
- IPerf started to run, collecting data.

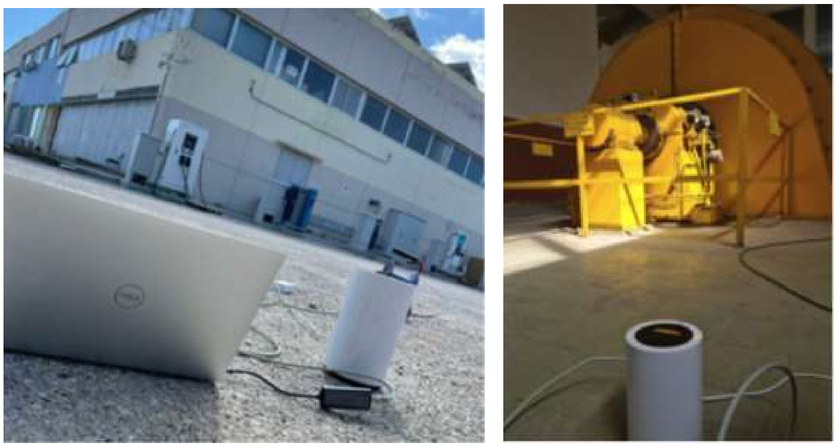
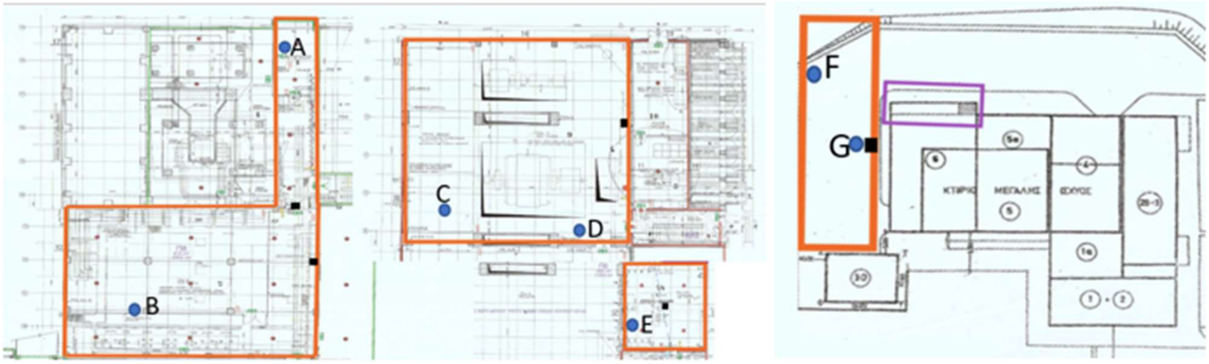


Figure 11 5G-gateway and end-user device used for network KPIs measurement.

CVM-01: measurement of the end-to-end delay from the time a packet leaves the user end device, until the reply gets back to the device. Can be extended to measure any delay between nodes in the network.

KPI value: [ms]



KPI	A	B	C	D	E	F	G
Latency (min)	5 ms	5 ms	4 ms	5 ms	4 ms	4 ms	4 ms
Latency (max)	9 ms	9 ms	8 ms	7 ms	10 ms	6 ms	6 ms
Latency (av)	6 ms	6 ms	5 ms	6 ms	6 ms	5 ms	5 ms

Remarks: Measurements are taken in several spots including several with obstacles between the antenna and the UE. From the results, it is shown that the network performance regarding latency can satisfy the requirements of UCs.

CVM04: number of application bits per second transferred over a specific use case interface.

KPI value: [Mbits/s]

KPI	A	B	C	D	E	F	G
Throughput (UL)	107 Mbps	103 Mbps	99 Mbps	98 Mbps	104 Mbps	107 Mbps	105 Mbps
Throughput (DL)	789 Mbps	866 Mbps	903 Mbps	837 Mbps	892 Mbps	873 Mbps	882 Mbps

Remarks: Measurements are taken in several spots including several with obstacles between the antenna and the UE. From the results, it is shown that the network performance regarding throughput can satisfy the requirements of UCs.

CVM05: number of data packets that were successfully sent out from one point in a network but were dropped during data transmission and never reached their destination, as a percentage of the total number of packets successfully sent.

KPI value: [%]

0% packet loss

Remarks: During the throughput and latency measurements, tens of thousands of packets were transmitted. The resulting conclusions are that packet loss reaches 0%, which leads to the calculation of availability and reliability at a rate that exceeds 99.9999%

CVM07: the radio access coverage area that provides the targeted application performance

KPI value: [m^2]

ground floor: 18.7x21 m

1st floor: 8.8x8.5 m

basement:18.7x21 m and 20x3 m

outdoor: 600 m2

Remarks: To cover the required multi-level area, 4 (indoor) radio dots and 1 outdoor antenna were installed. Figure 12 below shows the floor plans of the areas that were taken and the locations of the antennas.

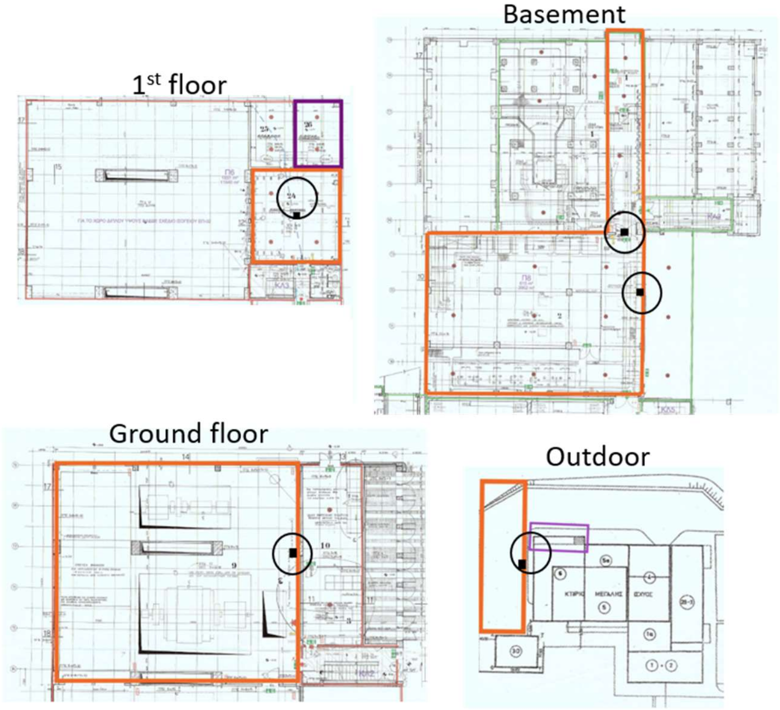


Figure 12 Area covered in ExFa-GR and the antennas location

6.3 ExFa-IT

Description

The ExFa network test consists of measuring and validating the network required KPIs in terms of **latency**, **throughput** and network **availability** and **reliability** for the 5G testbed. An additional KPI was introduced to ensure the full **compliance with health and safety** management policy in the industrial environment and related to the measurement of the electromagnetic field in the shop floor.

Th ExFa-IT demonstrator has been developed and tested with 2 different 5G connectivity options, with different type of engagement of the various UCs applicable to Whirlpool's facility:

1. 5G connectivity through Wind3 public network

   Infrastructure KPIs measurements are extended from the user devices in the shop floor, running the UCs specific nApps, to the core, hosting both the NAO orchestrator and UCs applications backend @ CNIT data Center, through the full connectivity lane of WIND3 network pivoting on its Data Center. KPIs' measurements are performed for each single application in the shop floor environment.

   The option is applicable to all ExFa-IT use cases (UC4, UC6, UC7, UC8).

2. 5G connectivity through CNIT mobile testbed

   Infrastructure KPI measurements are extended from the user devices in the shop floor, running the specific UCs nApps, to the core, hosting the NAO orchestrator @ CNIT data Center, through the mobile testbed hosting LOCALLY the application backend, provided by CNIT and installed locally in the Whirlpool's shop floor. Also in this case, the KPIs' measurements are performed for each single application; however, they involve only UC7 and UC8.

Purpose: in both cases, the expected results should validate the optimal performance of the network, in order to support the identified business requirements.

Network Deployment: Option 1

All the use cases are deployed using mobile devices connected through WIND3 public network: dedicated SIM cards are configured to ensure devices connectivity and 5G gateways are installed in the shop floor demo area. WIND3 public network ensures 5G connectivity to CNIT DevOps testbed hosted on a dedicated Data Center in Settimo Milanese (2 co-located servers: (1) SGW-PGW for core network at the edge and (2) APP server). The public APN is configured to connect SIM cards to SGW-PGW – naming of public APN that must be configured into device is "myinternet.wind".

SGW-PGW (server 1) is connected locally to server 2 but server 1 is protected by vFW, because server 1 is a part of WIND3 network and this vFW avoids potential security risks. Server 2 is connected locally to server 1 and to WIND3 IP Backbone in order to permit connection to CNIT. A VPN is configured between the firewall in WIND3 IP Backbone and CNIT. A vFW is managed by WIND3 Operations Dept, while another firewall in WIND3 IP Backbone is already managed by Op. Dept. The rules are configured in both firewalls for permitting required traffic.

A public IP address is assigned to share a VPN lane with CNIT.

SIMs' configuration: 1 SIM is configured with private APN 5G-INDUCE and is directly connected to the MEC; as WIND3 has already indicated in the past, this can only send traffic to the MEC and do not have access to the Internet. It has also APN internet.it to guarantee connection to the Internet without going through the MEC using a private natted IP address. Furthermore, 5 SIMs are configured with APN myinternet.wind, which allows, as requested in the past, to be able to transit directly on the Internet using a public IP address. They have also APN internet.it to guarantee connection to the Internet without going through the MEC using a private natted IP address.

Testing Procedure

Step 1: Install the two 5G gateways in the shop floor to support UC#7 and UC#8 demonstration execution.

Step 2: Equip both the 5G gateways and all mobile devices* with Wind3 SIMs, in order to allow UC#x devices registering into Wind 3 PUBLIC network and reaching backend of nAPPS/NAO, running on remote DevOps testbed.

*UC8 is replicating exacting the same probe mechanism both on fixed and mobile device, allowing flexibility on executing network performances monitoring everywhere (including flying drones, not tested in our industrial environment, due to privacy restrictions)

Step 3: Proceed with generic local measurements: 5G coverage, signal power, electromagnetic exposure, web reachability, download/upload throughput

Step 4: Verify the proper UCx nAPP operativity

Step 5: Every UC#s owner did proceed with project specific end-2-end KPI measurements along the different pieces of the solution (nAPP front-end/GWs → nAPP back-end/NAO), as described below

CVM01: Latency - measurement of the end-to-end delay from the time a packet leaves the user end device, until the reply gets back to the device. Can be extended to measure any delay between nodes in the network.

KPI value: [s] 2.051 ms

Remarks: The above value accounts for the end-to-end delay between the MEC server and the 5G Core of the Wind3 network. Measurements on the radio part not available.

CVM05: Packet Loss - number of data packets that were successfully sent out from one point in a network, but were dropped during data transmission and never reached their destination, as a percentage of the total number of packets successfully sent.

KPI value: [%] 0%

Remarks: No packet loss detected.

CVM06: Availability - The amount of time the end-to-end application is properly delivered according to the specified performance metrics, over the amount of time the that is expected to deliver the end-to-end nApp service.

KPI value: [%] 100%

Remarks: Result achieved after 5G gateway relocation nearer to windows (leveraging on outdoor network coverage)

CVM07: Coverage - the radio access coverage area that provides the targeted application performance

KPI value: [m^2] The whole area of Whirlpool’s factory is covered by Wind3 5G network even if with variable performances depending front he positioning of the 5G antennas

Remarks: The indoor coverage is not uniform in the demo area and the 5G gateways had to be moved nearer to external walls/windows after a measurement session held by Wind3. The coverage seems strongly affected by metal infrastructure present in the shop floor while the outdoor coverage ensures stronger signal.

CVM11: Health & Safety - compliance with national (D.Lgs 81/08, D.Lgs.159/16) and European regulation related to electromagnetic fields exposition, based on the calculation of the power density as power per area .

KPI value: [W/m^2] < 0.10 - full compliance

Remarks: the assessment of electromagnetic exposure risk has been executed on all devices used for the 5G-INDUCE demo experimentation, referring to a wide band measurement (100 kHz - 7 GHz) and reference 6 min of exposure.

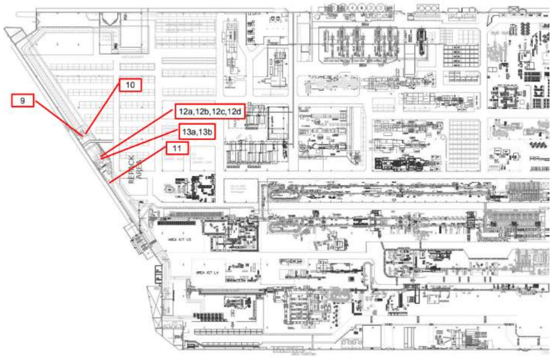


Figure 13 Positioning of the measurements executed in the shop floor (option1)

Figure 14 Measurement sessions (Option1)

<u>Network Deployment</u>: Option 2

UC7 and UC8 will be deployed using CNIT mobile testbed, which is composed as it follows.

SIM have been specifically programmed by CNIT in order to be able to register on the 5G network published by the MOBILE TESTBED described below

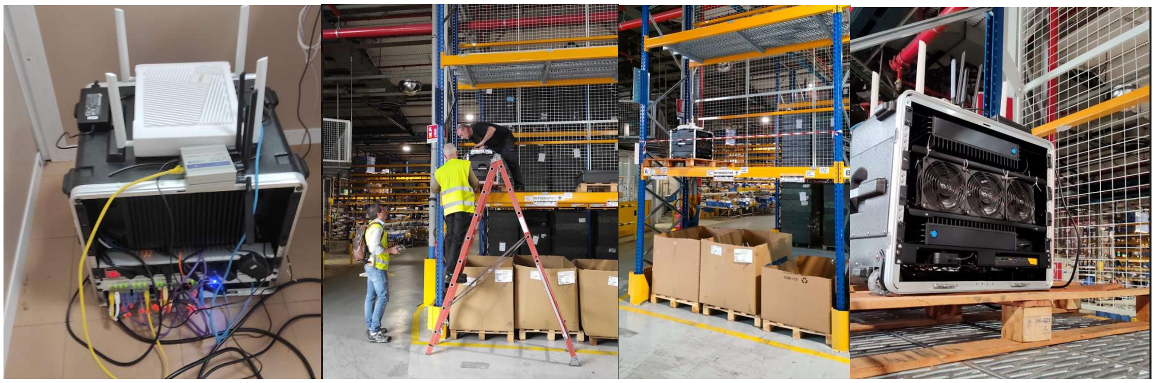


Figure 15 Mobile Testbed installation in the shop floor

The MOBILE TESTBED has been assembled and tested at CNIT labs in Genova and then transferred and installed in Whirlpool’s Refrigeration factory, near the demo area dedicated to UC7 testing, in order to ensure the connectivity of 5G gateways and 5G devices to be used for the UCs. Due to the limited range of the 5G microcell (10-15mt max), the demo of UC4 and UC8 will be executed only in this area instead of the other involved spaces in the shop floor (thermoforming dept. for UC4 and stamping dept for UC8) with no impact on the meaningfulness of measured KPIs. Moreover, the onsite physical installation has been complicated by the need to protect the testbed in the production area: to reduce the risk of collision and damages the device was installed on a higher position on a shelf and the location has been signed as not usable by material handlers.

Testing Procedure

Step 1: Replace Wind3 SIMs with SIMs configured by CNIT in the UC#7 & UC#8* 5G gateways, in order to allow UC#x infrastructure registering into CNIT MOBILE TESTBED, recahing backend of LOCAL nAPPS and connecting to the remote NAO

**as we experience issues on connecting this component directly to CNIT MOBILE TESTBED, we went indirectly through a portable 5G router (capable of registering to MOBILE TESTBED 5G network)*

Step 2: Equip test mobile device as well with SIMs configured by CNIT in order to allow UC#x devices registering into CNIT MOBILE TESTBED and reaching backend of nAPPS.

Step 3: Proceed with generic local measurements: 5G coverage, signal power, electromagnetic exposure, web reachability, download/upload throughput

Step 4: Verify the proper UC#x nAPP operativity

Step 5: Supported any request of configuration/restart of CNIT MOBILE TESTBED componenst

Step 6: Every UC#s owner did proceed with project specific end-2-end KPI measurements along the different pieces of the solution (nAPP front-end/GWs → nAPP back-end → NAO), as described below

CVM01: Latency - measurement of the end-to-end delay from the time a packet leaves the user end device, until the reply gets back to the device. Can be extended to measure any delay between nodes in the network.

KPI value: [s] 25.8 ms

Remarks: The above value accounts for the round-trip end-to-end delay, including the radio part and traversal of the VPN between the mobile testbed and CNIT premises in Genoa.

CVM05: Packet Loss - number of data packets that were successfully sent out from one point in a network, but were dropped during data transmission and never reached their destination, as a percentage of the total number of packets successfully sent.

KPI value: [%] 0%

Remarks: No packet loss detected.

CVM06: Availability - The amount of time the end-to-end application is properly delivered according to the specified performance metrics, over the amount of time the that is expected to deliver the end-to-end nApp service.

KPI value: [%] 50%

Remarks: the results are currently poor due to frequent shutdown of the connectivity which cannot ensure a stable connectivity required by the UC's (in particular UC7, where availability below 100% may affect safety).

CVM07: Coverage - the radio access coverage area that provides the targeted application performance

KPI value: [m^2] 100m^2 max

Remarks: The coverage is limited by design of the microcell assembly.

CVM11: Health & Safety - compliance with national (D.Lgs 81/08, D.Lgs.159/16) and European regulation related to electromagnetic fields exposition, based on the calculation of the power density as power per area .

KPI value: [W/m^2] < 0.10 - full compliance

Remarks: the assessment of electromagnetic exposure risk has been executed both on person level and on device level, always referring to a wide band measurement (100 kHz - 7 GHz) and reference 6 min of exposure.

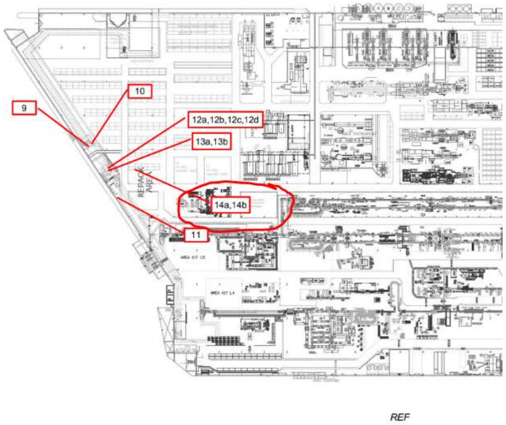


Figure 16 Positioning of the measurements executed in the shop floor (option2)

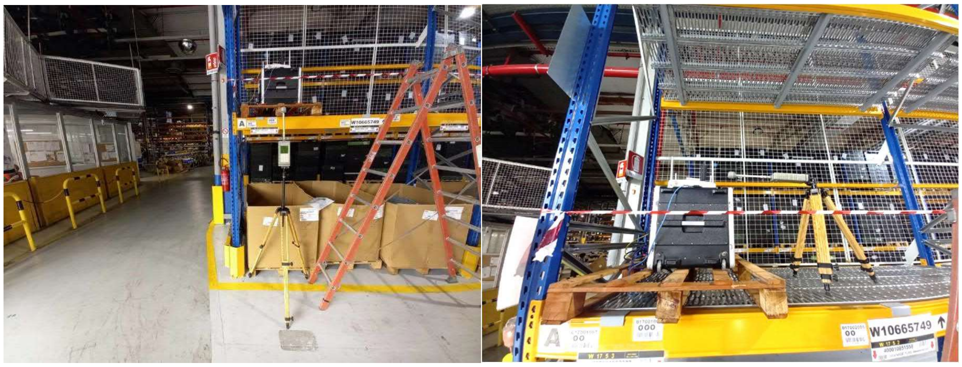


Figure 17 Measurements session (option2)

7 Conclusions

We have presented a detailed description of the DevOps testbed's structure and capabilities, along with a validation of the 5G-INDUCE platform readiness and performance, of the network applications onboarding and deployment, and the verification of the ExFas' readiness.

The DevOps infrastructure has been used to derive Core Validation Metrics (CVMs) regarding the verification of NAO-OSS interworking, as well as the evaluation of the 5G-INDUCE Platform performance with regard to the deployment of nApps. In particular, the CVM regarding interworking of the NAO and OSS components of the 5G-INDUCE platform (time elapsed from the triggering of the creation of a slice through the NAO slice intent module to the OSS until the completion of nApp deployment by NAO, once the slice is fully established and operational), has been evaluated on both the DevOps testbed at CNIT premises and the 5TONIC/Ford/UPV ExFa that spans over the 5TONIC project's, Ford's and UPV's premises in Madrid and Valencia, Spain, in order to compare deployment operations over two significant alternative facilities.

The DevOps testbed has been also employed to perform the onboarding and deployment validation of the nApps regarding the various use cases, to obtain a comparison over a uniform environment. Finally, the readiness of the ExFas and their ability to host the integrated 5G-INDUCE solution has been tested in the three specific environments in Spain, Greece and Italy, providing results in accordance with the foreseen capabilities. The final validation of the Use Cases will be the subject of deliverable D6.2.

References

[1] "Slices-SC - Scientific Large-Scale Infrastructure for Computing/ Communication Experimental Studies." [Online]. Available: https://slices-sc.eu/.

[2] "ESFRI - European Strategy Forum on Research Infrastructures." [Online]. Available: https://www.esfri.eu/.

[3] P4 Language Consortium, [Online]. Available: https://p4.org/.

[4] HUAWEI 5G CPE Pro 2, [Online]. Available: https://consumer.huawei.com/en/routers/5g-cpe-pro-2/

[5] AMARI UE Simbox Series, [Online]. Available: https://www.amarisoft.com/products/test-measurements/amari-ue-simbox/

[6] Apache Guacamole, [Online]. Available: https://guacamole.apache.org/

[7] SCRCPY: An Android Screen Mirroring Tool, [Online]. Available: https://scrcpy.org/

[8] GARR - the Italian Research and Education Network, [Online]. Available: https://www.garr.it/en/

[9] R. Bolla *et al*., "A Multi-Tenant System for 5/6G Testbed as-a-Service," 2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 2023, pp. 768-773, doi: 10.1109/COMSNETS56262.2023.10041360.

[10] Canonical MaaS, [Online]. Available: https://maas.io/

[11] RedHat Ansible - Drive automation across open hybrid cloud deployments, [Online]. Available: https://www.ansible.com/

[12] Prometheus, [Online]. Available: https://docs.openstack.org/openstack-helm-infra/latest/ monitoring/prometheus.html.

[13] Elastic Stack - Meet the search platform that helps you search, solve, and succeed, [Online]. Available: https://www.elastic.co/elastic-stack.

[14] ETSI, ETSI GS NFV-SOL 006 V2.7.1, "Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; NFV descriptors based on YANG Specification", December 2019.