Grant Agreement No: 101016941

Full Title: Open cooperative 5G experimentation platforms for the industrial sector NApps

Start date: 01/01/2021

End date: 30/06/2024

Duration: 42 Months

Project URL: https://www.5g-induce.eu/

Deliverable D3.6

The 5G-INDUCE orchestration framework platform

Document type	Deliverable		
Title	D3.6 – The 5G-INDUCE orchestration framework platform		
Contractual due date	31/12/2023 (M36)	**Actual submission date**	19/01/2024
Nature	Report	**Dissemination Level**	Public
Lead Beneficiary	CNIT		
Responsible Author	Chiara Lombardo (CNIT)		
Contributions from	Maurizio Giribaldi (INFOCOM), Dimitris Klonidis, Thanos Xirofotos (UBITECH)		

Revision history

Version	Issue Date	Changes	Contributor(s)
v0.1	09/11/2023	First draft.	Maurizio Giribaldi (INFOCOM) Chiara Lombardo (CNIT) Thanos Xirofotos (UBITECH)
v0.2	10/01/2024	All contributions have been collected.	Maurizio Giribaldi (INFOCOM) Chiara Lombardo (CNIT) Thanos Xirofotos (UBITECH)
v0.3	18/01/2024	Introduction and conclusions have been added; final content revision; ready for submission.	Franco Davoli (CNIT) Maurizio Giribaldi (INFOCOM) Chiara Lombardo (CNIT) Thanos Xirofotos (UBITECH)
v1.0	19/01/2024	Final formatting revision.	Riccardo Rapuzzi (CNIT)

Table of Contents

List of Figures

Glossary of terms and abbreviations used

Abbreviation / Term	Description
3GPP	Third Generation Partnership Project
API	Application Programming Interface
B5G	Beyond 5G
CRUD	Create, Read, Update, and Delete
DevOps	Development and Operations
eMBB	enhanced Mobile Broadband
eNodeB	E-UTRAN Node B
ETSI	European Telecommunications Standards Institute
ExFa	Experimentation Facility
gNodeB	Next-Generation Node B
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
ICCID	Integrated Circuit Card Identifier
IPMI	Intelligent Platform Management Interface
LLDP	Link Layer Discovery Protocol
MaaS	Metal as a Service
MANO	Management and Orchestration
mMTC	massive Machine Type Communications
NAO	Network Application Orchestrator
nApp	Network Application
NB	North-Bound
NFV	Network Function Virtualization
NFVCL	NFV Convergence Layer
NFVO	NFV Orchestrator
OSM	Open-Source MANO
OSS	Operation Support System
PaaS	Platform as a Service
PDU	Protocol Data Unit
PNF	Physical Network Function
QoS	Quality of Service

Abbreviation / Term	Description
RBAC	Rule-Based Access Control
REST	Representational State Transfer
RRH	Remote Radio Head
SB	South-Bound
SLA	Service Level Agreement
UE	User Equipment
URLLC	Ultra Reliable Low Latency Communications
VAO	Vertical Application Orchestrator
VIM	Virtual Infrastructure Manager
VLAN	Virtual Local Area Network
VNF	Virtual Network Function
VNFD	Virtual Network Function Descriptor
YAML	YAML Ain't Markup Language

Executive Summary

This document presents the finalized version of the 5G-INDUCE platform, including its components, all the developed features, and outlines some integration aspects within the platform and with heterogeneous infrastructures.

In particular, this deliverable represents a summary of the progress previously reported in D3.4 and D3.5, in the form of a white paper to expose the platform outside the 5G-INDUCE project. In this respect, the deliverable describes the NAO, the OSS and their interactions to manage the lifecycle of Network Applications (nApps) onto 5G and beyond (B5G) infrastructures characterized by different levels of programmability.

The goal of the current document is to disseminate the completed 5G-INDUCE platform to a wider audience, as D3.4 and D3.5 were private documents and highly technical ones. For this reason, the current deliverable provides a more high-level view of the platform, glossing over specific development details but still providing a substantial level of knowledge on the most relevant features and interactions, in order to appreciate the innovations introduced by 5G-INDUCE for the lifecycle management of nApps and related slices.

1 Introduction

1.1 Deliverable Purpose

This deliverable summarises the content of D3.4 and D3.5 and presents the 5G-INDUCE orchestration platform in the form of a white paper, of which the next section can be considered the introduction. The goal of the current document is to provide readers outside of the project consortium (as it is the only WP3 public deliverable) with a high-level understanding of the building blocks, the functionalities and the integration requirements of the platform.

1.2 Relation with other Deliverables and Tasks

This deliverable represents the final outcome, i.e., the release of the 5G-INDUCE platform, achieved in the scope of the WP3 tasks, namely:

- T3.1 “5G-INDUCE Network Application Orchestration Development”,
- T3.2 “5G-INDUCE Operations Support System Orchestration development”,

and

- T3.3 “5G-INDUCE open platform integration and interfacing”.

In accordance with the implementation strategy and plan described in the Grant Agreement [1], WP3 has produced three incremental versions (Version A to C) of the 5G-INDUCE Orchestration Platform, which correspond to the deliverables D3.1, D3.2 and D3.3. Furthermore, WP3 has delivered two further documents, namely deliverables D3.4 and D3.5, which provided the key design features and functionalities of the 5G-INDUCE Platform Version A and C, respectively.

While D3.5 was an evolution of D3.4, capturing the updates and modifications of platform features and design intervened in between the releases D3.2 and D3.3, the current document still focuses on Version C but it targets a wider audience, and as such descriptions are more high-level, focusing on the functionality rather than on the implementation aspects.

1.3 Deliverable Structure

The remaining of the document represents the publishable description of the 5G-INDUCE platform in the form of a white paper. Hence, Section 2 provides the introduction to the platform, while Sections 3 and 4 describe the NAO and the OSS, respectively. Section 5 covers integration aspects, both between NAO and OSS and towards the Experimentation Facilities (ExFas); conclusions can be found in Section 6.

2 The 5G-INDUCE Platform

The 5G INDUCE platform is specifically conceived for simplifying and automating the management of Network Applications (nApps) onto 5G and beyond (B5G) infrastructures. At a glance, the proposed platform aims to mostly hide the complexity of the 5G environment to application developers and providers and make the development, deployment and operation of 5G-ready nApps similar to the well-known corresponding processes applied to cloud-native applications in cloud computing environments.

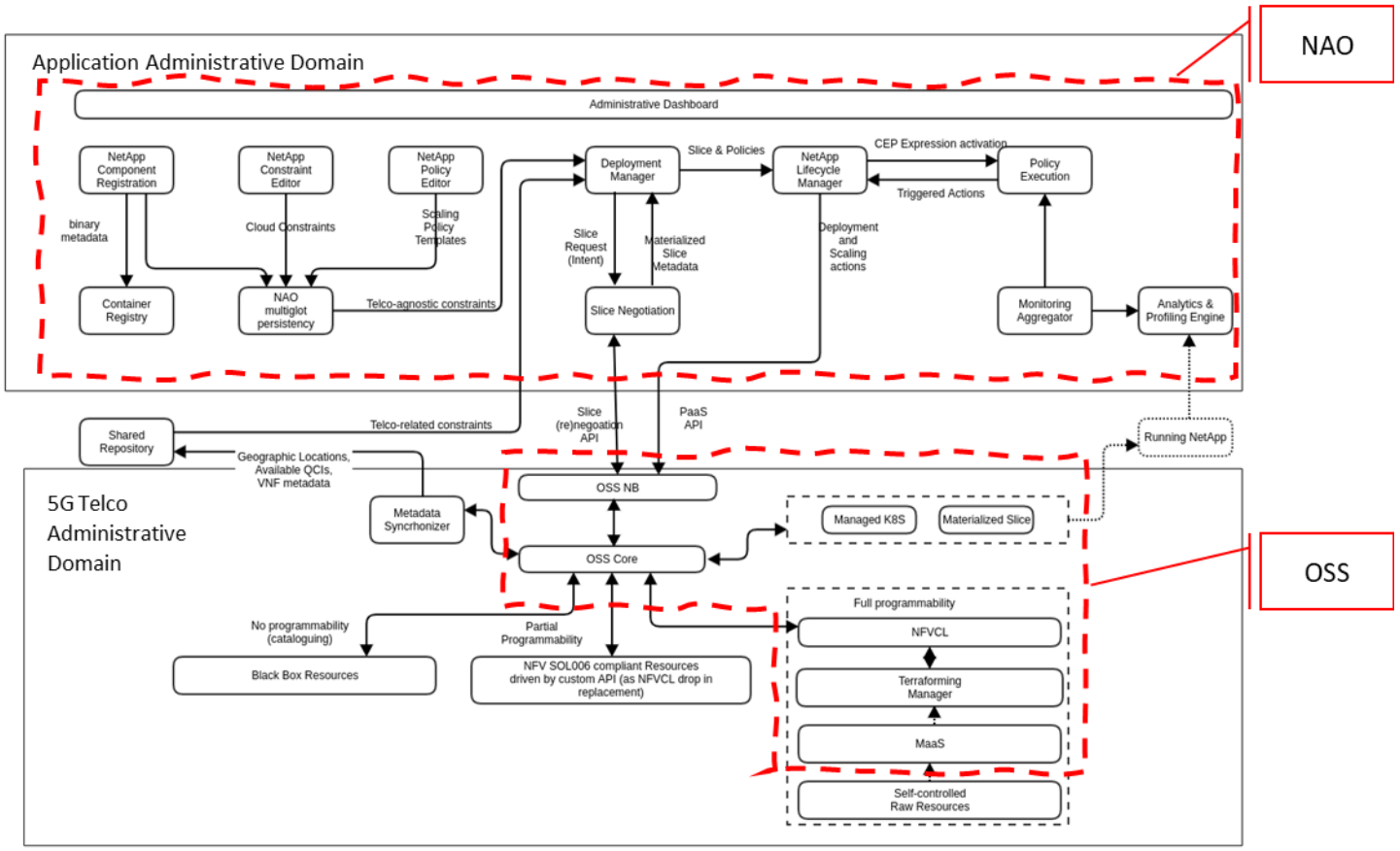

Figure 1: The 5G-INDUCE architecture.

Figure 1 depicts the general 5G-INDUCE conceptual architecture and highlights the two 5G-INDUCE platform components, whose features and implementations are extensively described in the following parts of this deliverable: the Network Application Orchestrator (NAO) and the Operation Support System (OSS).

The split of the 5G-INDUCE orchestration platform functionalities between the two aforementioned components matches the need to ensure the so-called “separation of concerns” between the two different administrative domains the platform spans; namely: the Application Domain and the Telco Domain. According to this paradigm, the lifecycles of nApps and network services have to be managed by separated orchestration tools; each orchestration tool is devoted to its specific (applicative or network) administrative domain, which is owned and operated by a specific stakeholder, such as, for instance, a vertical industry, for the Application domain, and a 5G network operator, for the Telco Domain.

Such orchestrators obviously need to strictly cooperate to allow the coherent use and configuration of computing, storage and network resources and ensure the correct end-to-end deployment of a 5G-ready

application along with the supporting network services. Specifically, the interworking between NAO and OSS is realized through an intent-based interface, called “5G and edge computing slice negotiation interface”. Upon nApp deployment requests, through the slice negotiation interface, the NAO can request, negotiate and obtain from the OSS the needed computing resources at the edge facilities where to run nApp components, as well as the networking services to ensure the connectivity among such resources and with the User Equipment (UE). The OSS has the task of analysing operational and performance (soft and hard) constraints expressed by the NAO slice request, and, consequently, selecting the most suitable computing facilities and network services complying with the requirements. The initial request produced by the NAO towards the OSS is called “slice intent”; it includes the application graph, annotated with QoS and operational requirements. Following the successful completion of the negotiation, the OSS provides back to the NAO the “materialized slice”, which describes the set of needed network services and resources, instantiated and configured, ready to support the nApp.

Moreover, the NAO-OSS interface supports the modification and reconfiguration of the slice within its lifecycle and enables advanced operations to deal with UE mobility and dynamic QoS/operation requirements. In more detail, 5G-INDUCE nApps (or some of their components) can be lively scaled and relocated onto computing facilities in new geographical areas or at different network infrastructure aggregation levels in a transparent and smooth fashion. nApp traffic from and to UEs is steered accordingly (by coherently updating the configuration of network slices and related network services), while the platform adapts the nApp geographical scope and properly scales the components and network services on each area depending on the number of hosted nApps, the local workload, and the dynamic QoS/operational requirements.

For instance, live updates on the bandwidth/latency requirements of a link in the nApps graph notified by the NAO might trigger the OSS to perform scaling, reconfiguration or relocation of some of the involved VNFs. Updates on nApp component scaling can trigger the OSS to interact with edge computing facilities to correctly re-dimension the allocated resources. A second explanatory example is related to the geographical scope of a nApp deployment, which is the geographical area covered by the radio mobile network from where UEs can access the nApp and the requirements expressed in the slice intent are met. 5G-INDUCE will add to the NAO the capability of dynamically requesting to live change the geographical scope of a running nApp instance (i.e., for coping with UE mobility or for administratively extending/resizing the working area of the nApp). This geographical scope modification, on one side, entails the capability of the OSS to select/deselect proper resources at edge facilities, by updating/creating/deleting network services, and to dynamically reconfigure network slices to transparently/smoothly redirect UE incoming/outgoing traffic during the reconfiguration phases. On the other hand, through a proper synchronization with the OSS, the NAO can update the nApp instance graph by deploying/removing nApp components in the selected edge facilities. In other words, the NAO will be equipped with a new function for the “right/left scaling” over the geographically distributed edge infrastructure.

A further main innovation point of the 5G-INDUCE OSS is that it allows to flexibly handle any NFV virtualization levels, which means that network services can be composed by mixing VNFs that are realized not only with IaaS resources, but also with cloud-native containers over PaaS platforms (e.g., 3GPP 5G core network functions over Kubernetes clusters), and physical programmable/configurable devices (e.g., gNodeBs, eNodeBs, etc.). The number and the types of network services will be extended by including cloud-native 3GPP 5G network functions and zero-touch procedures to enable the dynamic management of slices and traffic steering.

In this respect, the OSS includes a special module; namely, the NFV Convergence Layer (NFVCL), which fully drives NFV service orchestration along all the lifecycle phases. NFVCL communicates with an external NFV

Orchestrator (e.g., ETSI Open-Source MANO – OSM) through standard ETSI NFV interfaces. During Day-0 operation, NFVCL produces and onboards the ETSI SOL006 [2] descriptors of services and of related Virtual/Physical/Container Network Functions 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, through a Metal-as-a-Service (MaaS) [3] approach, the OSS also provides the capability to manage and to terraform bare-metal resources (i.e., servers, switches, routers, etc.) to install, build, and configure complex and distributed IaaS and PaaS environments, where to host VNFs and nApps components.

The 5G-INDUCE platform has been specifically designed to be flexibly adaptable to the programmability level offered by the underlying network infrastructure(s) by enabling the aforementioned capabilities when and where possible.

3 The 5G Network Application Orchestrator (NAO)

The 5G-INDUCE NAO is the upper part of Figure 1 and is responsible for the functionalities of registering a nApp and all its components, authoring the application deployment and runtime policies, providing application monitoring, negotiating slices and managing the operational state of the nApp within the scope of an allocated slice. The NAO interplays with the lower part of Figure 1, the OSS, through a slice intent-reply mechanism for the deployment and run-time management of the nApp lifecycle in order to dynamically enable, scale and modify nApp capabilities according to relevant events e.g., the bandwidth of the radio link, the scaling of the nApp deployment including alternating the respective required resources, etc.

The NAO provides the interfacing layer with the end user (i.e. the vertical industry expert and the application developer) for managing the deployable applications and their features. Figure 2 depicts the internal to the

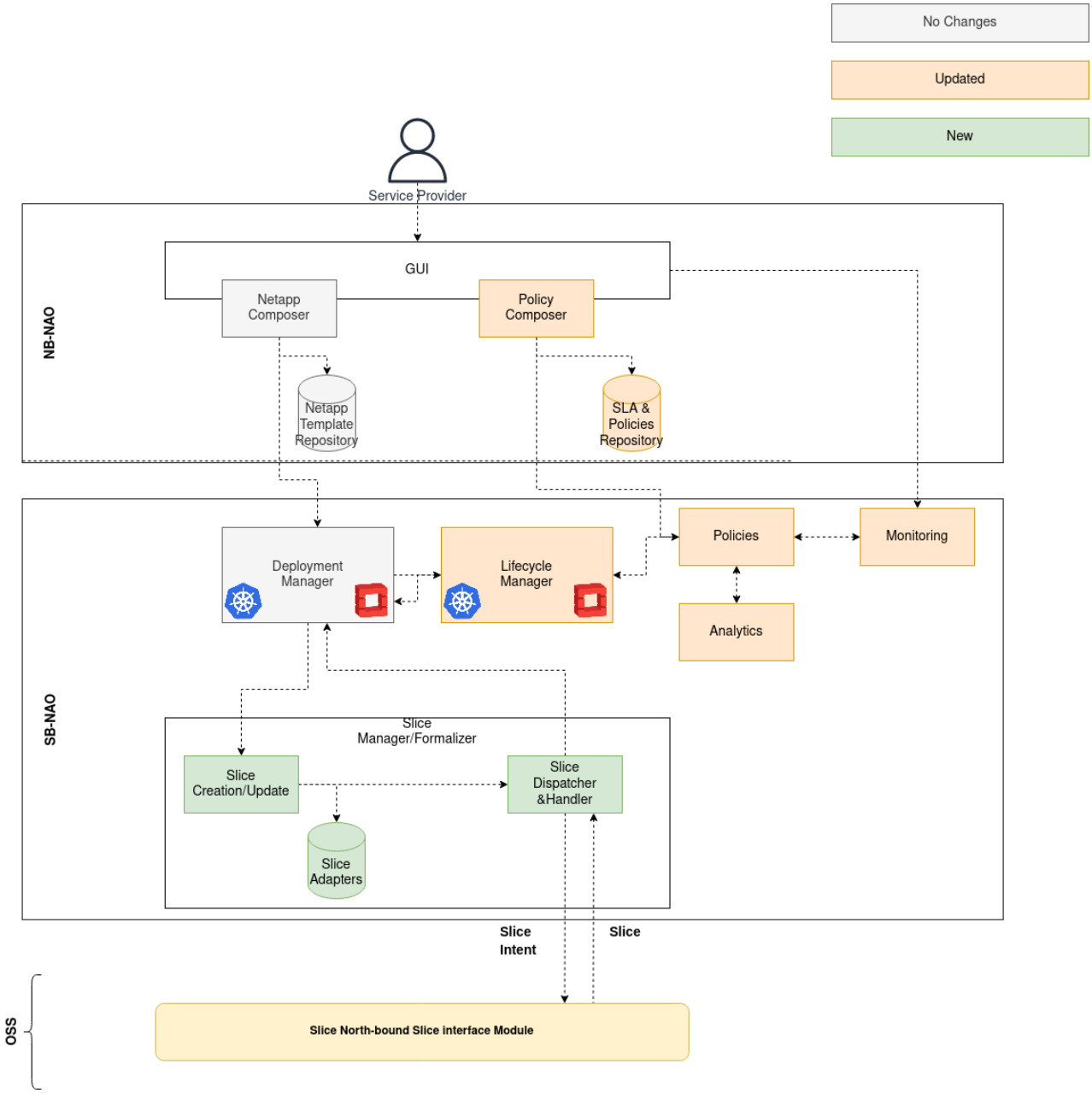

Figure 2: Overview of the 5G-INDUCE NAO.

NAO architecture, highlighting the updates on a per module basis. The latest developments are the **updated** submodules, i.e., **monitoring, policies and analytics** (in orange) and they are related with the runtime functionalities of a nApp. Similarly, the new developments i.e., **slice update, slice dispatch and handler** (in green) are related with the slice update feature of the 5G-INDUCE Platform. The arrows that are marked as **slice intent/slice** are the actions supported by the interface with the OSS.

The internal modules of the NAO in Figure 2 are grouped into the North-Bound (NB) and South-Bound (SB) modules. The NB modules of the NAO provide the interface with end users, specifically, nApp registration, creation and management, nApp monitoring and policies editing. For these functionalities, there are dedicated Graphical User Interfaces (GUIs) that combine several panels, in order to project the collected information from the deployment and lifecycle of every deployed nApp. The SB modules of the NAO are the "back-end" for the functionalities of the NB plus the orchestration functionalities that are the initial deployment of an application along with its lifecycle operations to maintain the operational state of it. Additionally, the creation and reception of the slice from the OSS are also SB functionalities of the NAO.

3.1 The NAO North-Bound Services

3.1.1 nApp Composer

The nApp Composer helps the service providers to wrap the containerized application in a proper format, so as to be publishable in the nApp Template Repository. Each nApp consists of multiple containers that are chained in the form of a service graph. The composition process guarantees that the requirements for deploying each nApp are obtained from the end user and stored as metadata. These metadata are regarding *minimum infrastructural requirements*, metadata regarding *deployment preferences*, metadata regarding configuration *parameters during component initialization*, mutable configuration parameters during runtime, exposed and required interfaces, exposed metrics. There are also metadata related to the required QoS that play a significant role in the slice negotiation process that are passed to the OSS through the slice intent. These network metadata are related with:

- the link quality between the components of the nApp, the required radio capabilities (eMBB (enhanced Mobile Broadband), URLLC (Ultra Reliable Low Latency Communications), mMTC (massive Machine Type Communications));
- the QoS Class Identifier classification;
- the minimum guaranteed bandwidth;
- the maximum bit rates;
- list of UEs that are allowed to access the slice;
- geographical Area for the deployment.

3.1.2 Policy Composer

For the nApps that are deployed, runtime changes/reconfigurations can be realized. This reconfiguration aims to the satisfaction of a set of business goals that are bundled in the form of a Service Level Agreement (SLA). In the frame of satisfaction, there are specific control plane "actions" that are supported. Specifically, the supported actions are:

a. the update (scale up or scale down) of the computing resources, in order to overcome computational bottlenecks of the nApp,
b. the update of networking resources the slice embeds, which can be scaled according to the evolving requirements of the deployed nApps.

The registered policies are targeting specific deployed nApp and are provided by the end user. The user can set and store (in Figure 2 the SLA & Policy Repository) the policy rule to be fired upon some runtime value

that is coming from the monitoring. For every active policy rule, the triggered action that can be performed is realized through a slice-update mechanism.

3.2 The NAO South-Bound Services

3.2.1 Monitoring

The monitoring mechanism is using agents for collecting metrics and publishes them according to the producer-consumer paradigm [4]. The concept is the following: every nApp component that is deployed from the 5G-INDUCE Platform is wrapped "inside" a Kubernetes pod or a Virtual Machine (VM, depending on the virtualisation technology that the underlying infrastructure provides) alongside with a 5G-INDUCE Agent-Sidecar and a 5G-INDUCE Monitoring Agent. 5G-INDUCE Agent-Sidecars are responsible for *i*) for building the execution environment, *ii*) monitoring the boot sequence and *iii*) reporting the operational state, and, finally, *iv*) **for starting and configuring the Monitoring Agents for every component.** The Monitoring Agents are the entities responsible for setting up the metric collection process by activating and configuring a set of Monitoring Probes. The Monitoring Probes can be enabled from metric acquisition and can obtain: *i*) hardware and kernel-related metrics, *ii*) containers and applications, *iii*) service availability metrics, and *iv*) custom metrics that the user needs to provide code for extracting and publishing this to the monitoring probe.

Additional functionality of Monitoring Agents includes adapting the intensity of the monitoring process by changing the periodicity of both the metric collection and dissemination process implemented by probes, based on the evolution of metric data stream (adaptive monitoring).

3.2.2 Policies & Analytics

Run-time policies' management is realized through policy rules, which are configured by the 5G-INDUCE platform users and administrators. The concept of policies follows a triple stage approach where the policies' rules are 1) created,2) processed, and 3) when the condition of the rule is being validated there is an associated action that is being triggered. The form of creating rules is provided through a GUI and the current version of the Policy management system allows policy rules to be submitted and deleted at any time for the running nApps.

A user can utilize a GUI to submit requests for the creation or configuration of policy rules. Through the same GUI, the user can receive pertinent alerts and log messages upon the activation and deactivation of various rules. In the rule creation stage (Stage 1) a list of nApp components along with associated metrics that comes from the Monitoring Engine and an action is being used to compose a policy rule. Once the rule parameters are defined, the next step (Stage 2) is to transform them in a comprehensive format expressed in PromQL (Prometheus Query Language [5]) and to outline the condition that, when met, triggers an alert. Subsequently, based on the transformed rule, a corresponding alert is being generated and placed in the Monitoring Engine. In this stage, the formulated rule is being transformed in a YAML descriptor and deposited into a specific repository within the Monitoring Engine.

In the final stage (Stage 3), there is a continuous lifecycle management of the registered policy rules. This involves more than just storing and indexing rules to monitor their status. More precisely, in this stage real-time updates about the status of each rule are continuously collected, in order to evaluate when a rule is activated and, as a consequence of this, to trigger the associated action. The associated actions that can be triggered by a policy rule are targeting mainly the resource scaling such as increase/decrease the number of nApp component instances that also involves the scaling of the environment (namespace, tenant) where the nApp runs, so to be able to host the new number of instances (in the case of scale up). In general, this stage bears the responsibility for triggering actions in the event of a rule violation and disseminating pertinent information to facilitate the execution of subsequent actions by the Platform.

3.2.3 Deployment Manager

The Deployment Manager carries out the task to realise the initial deployment of a nApp Graph on top of the programmable resources by undertaking the task of communication with the underlying registered infrastructure. To do so, the Deployment Manager is fully aware of the available resources, their state, and directive-indications (declared through the 5G-INDUCE specific metadata described in high level in Section 3.1.1) for the actual deployment. The knowledge of the resource-state derives from the interaction of the Deployment Manager with the NB of the OSS (see Section 4.1). The functionality to perform deployments in various virtualization environments derives from a built-in capability to interact with various virtualization endpoints. Hence, the Deployment Manager provides an abstract interface for basic management virtualization operations (e.g., select/create tenant, create/modify namespaces, create/select credentials/secrets, obtain/select kubeconfig), by incorporating some of the industry-dominant virtualization technologies (OpenStack, Kubernetes) and their respective APIs. Finally, the Deployment Manager mainly interacts with the Slice Manager, so to send the requirements for a proper slice configuration or reconfiguration in the slice negotiation with the OSS (see section Slice Manager/Formalizer).

3.2.4 Lifecycle Manager

The Lifecycle Manager is a closed control loop that continuously assesses the existing resources, the deployment requests, and the reconfiguration requests of the already deployed nApps. With the Deployment Manager, they host and encapsulate the core orchestration business logic and can be seen as a centralized logical entity that affects the “scheduling” of nApp components that are deployed. In a nutshell, the Lifecycle Manager is responsible to maintain the operational state of the nApp. The initial state of nApp component is a tuple of requirements passed as metadata that are related with the number of resources that needs to be allocated, the slice parameters that have been selected, the geographical constraints. Any change in the previous requirements can happen during runtime (after the initial deployment) by the activation of policies. The Lifecycle Manager is responsible to trigger the slice (re)negotiation and after this step the control-plane signalling with the respective virtualized infrastructure API. The operational state of each nApp component is being monitoring and reported to the Lifecycle manager by the 5G-INDUCE Agent-Sidecars wrapped in every nApp component deployment for *i*) building the execution environment, *ii*) monitoring the boot sequence, *iii*) reporting the operational state, and *iv*) starting and configuring the Monitoring Agents for every component. The orchestration part of Lifecycle Manager includes specific management virtualization operations such as create/delete deployment, get deployment status, which are complementary to the Deployment Manager’s management actions. The Lifecycle Manager interacts mainly with the Deployment Manager, in order to perform a nApp deployment or un-deployment. For the the update (scale up or scale down) of the computing resources, the request for the slice renegotiation is being passed to the Slice Manager only through the Deployment Manager.

3.2.5 Slice Manager/Formalizer

The slice negotiation describes the interactions between the NAO and the OSS for the slice configuration. The slice negotiation includes a) the slice request phase, b) the slice instantiation, c) the slice operation and d) the slice deprovision. The Slice Manager is responsible for formulating the application slice request (intent) that contains all the **application requirements that come from the nApp Composer** (see Section 3.1.1) into a common format before sending it to the OSS. During the slice instantiation phase, these application requirements are translated into a slice configuration plan before being sent to the OSS. The Slice Manager is also responsible for receiving and parsing the realized slice from the OSS before a deployment starts. The translation of the requirements to a proper slice configuration plan, as well as the reception and extraction of the necessary information (i.e., connection details like IP, port, authorized user, kubeconfig, credentials like hashed key) from the realized slice before sending this to the Deployment Manager, is the primary objective of the Slice Manager.

4 The 5G Operation Support System (OSS)

The 5G-INDUCE Operations Support System (OSS), depicted in Figure 3, is in charge of managing all functions and operations required for the nApp placement over edge computing facilities and for its connection to a (properly configured) network slice, as well as maintaining the information on all the data regarding the deployed applications, network services, and available infrastructure resources.

The OSS is designed according to a highly modular architecture: all the software services are state-of-the-art cloud-native software, i.e., stateless services (or more precisely services with a state maintained in an external database; namely, MongoDB [6] and Prometheus [7]), inherently parallelizable. The 5G-INDUCE OSS architecture is organized in a suite of five main software services, grouped into two main modules: the North-Bound OSS (NB-OSS) and the South-Bound OSS (SB-OSS). The former module is meant to front-facing the NAO by managing slice negotiations for nApps, and to maintain metadata (e.g., coverage area served, operational capabilities, etc.) of one or multiple onboarded SB-OSS modules. The SB-OSS is meant once per each different administrative network/computing resource domain onboarded onto the OSS. To reflect the different programmability levels exposed by such administrative domains (e.g., the various ExFa testbeds), the SB-OSS has been designed as a chain of software services that can be selectively activated to gain access to various programmability levels, passing from a simple catalogue of available resources in case of no programmability, up to the complete terraforming of the physical infrastructure in case of full programmability. Even if not made explicit in the following, in order to maximize the flexibility of the approach, the SB-OSS will have the capability to maintain different programmability levels for edge computing and network (service/slice) resources in the same administrative domain.

In more detail, the NB-OSS is composed of two main services: the Slicing-Interface and the North-Bound Core services. The Slicing-Interface service is meant to implement the OSS APIs for the interface with the NAO. The NB Core service is in charge of two main tasks: onboarding SB OSS instances, and suitably process and propagate slicing requests/replies between the Slicing-Interface service (and then the NAO) and the relevant SB-OSS(es).

The SB-OSS will include three “chained” services: the South-Bound Core service, the NFV Convergence Layer (NFVCL), and the Metal Convergence Layer (MetalCL). The South-Bound Core service is the only mandatory element in the SB-OSS, and it is devoted to process the slice instantiation/modification/de-instantiation requests and related resources. This service is the key component for providing adaptive programmability: if the NFVCL and the MetalCL services are available, the SB Core can request them the setup or the change of new or existing network slices/services and of infrastructure resources (e.g., of OpenStack VIM instances and of the hosting physical servers). In case that bare-metal or virtualization programmability levels in an administrative domain are not exposed to the 5G-INDUCE platform, the SB-Core can dynamically request an external NFV framework for the needed slices/configurations, or simply cataloguing the pre-configured resources (e.g., a 5G network slice) statically dedicated to the 5G-INDUCE platform.

The role of the NFVCL within the SB-OSS is to manage the lifecycle of NFV services to provide suitable connectivity to nApp components and UEs in fully automated and zero-touch fashion. If not provided by the bare metal layer, the NFVCL is also in charge of providing and maintaining cloud-native computing frameworks at edge facilities (i.e., realizing Kubernetes clusters as NFV services).

The MetalCL is the service dedicated to managing and terraforming bare-metal resources (i.e., physical servers and hardware network equipment) to create IaaS/PaaS environments compliant with the 5G-platform needs. Also in such a case, this service allows the dynamic Day-0 to -N lifecycle management of operating systems in the servers, of configurations in network equipment, an of complex distributed applications like OpenStack and Kubernetes.

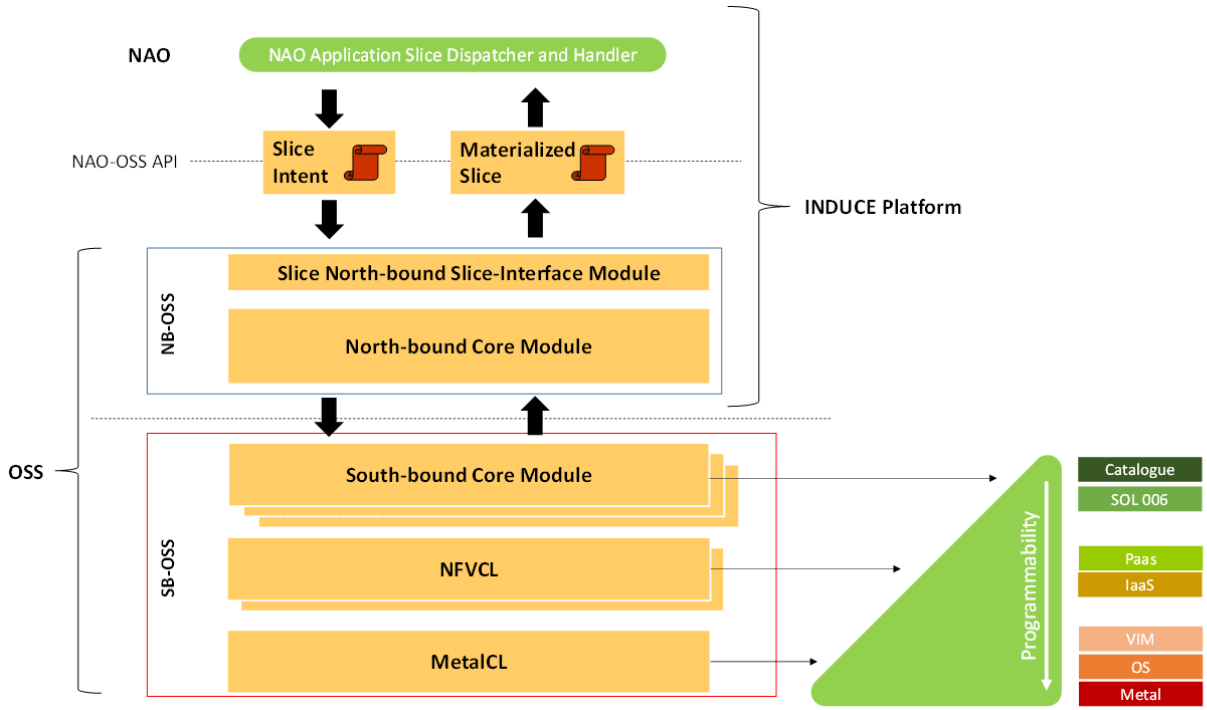

Figure 3: The OSS architecture.

4.1 The OSS Northbound Services

4.1.1 The Slicing Northbound Service

The Slicing Northbound service is responsible for the interaction between the NAO and the OSS. In particular, it oversees the messages to be exchanged in order to request the creation of a slice, including the computational, networking and QoS requirements, as well as for the slice lifecycle management up to its termination. It substantially corresponds to the OSS-side implementation of the NAO-OSS APIs that is described in detail in Section 5.1.

4.1.2 The NB Core Service

The OSS Core is split into two services, with a north-bound one statically included in the 5G-INDUCE platform and a south-bound one that will be plugged for each underlying administrative domain providing network and edge computing resources, but with possibly different programmability levels. In detail, it has been decided to maintain only the NB-OSS as part of the 5G-INDUCE part, because it represents the natural extension of the NAO in the Telecom Service Provider domain, whist the SB-OSS can be meant as an adaptation software stack to onboard heterogenous 5G infrastructures (and, therefore, it may fall at least partially into the domain of Telecom Infrastructure Providers).

The NB Core service maintains the original role of a high-performance session manager and message dispatcher based on the CRUD (Create, Read, Update, and Delete) paradigm. When a new slice intent request from the NAO is received, the OSS Core service parses it against the resources in the administrative domains made available by registered SB-OSSes.

The geographical scope and the PLMN identifier of the 5G network is retrieved and used to filter which SB-OSSes can be used to materialize the slice intent. If the slice geographical scope cannot be served by a single SB-OSS, multiple ones will be selected, and, if needed, the slice intent information will be properly split. If multiple SB-OSSes can be used to serve the same area, a policy defined by the operator will be used to select priorities among the filtered alternative SB-OSSes.

The NB-OSS passes through each area composing the scope, and it iterates over the possible alternatives by requesting the feasibility of the slice materialization. The first SB-OSS confirming the feasibility of the slice is chosen. At this point, the NB-OSS returns to the NAO (via the Slice-Interface service) the overall candidate for slice materialization (which will correspond to the super-position of the sub-slices coming from the different SB-OSS in the different geographical areas).

Upon the positive confirmation from the NAO, the NB-OSS asks the selected SB-OSS to materialize the slice and returns back a notification when the environment is ready.

Similar procedures are applied also in the case of live renegotiation of a slice, especially in the case of geo-scaling operations affecting the geographical scope of the nApp, and consequently triggering the possible selection or deselection of SB-OSSes.

4.2 The OSS Southbound Services

4.2.1 The SB Core Service

As explained in the beginning of Section 4, the SB Core service is the key element in the 5G-INDUCE architecture to handle different administrative domains with heterogeneous programmability levels, and more in detail:

- *No programmable domains*: in such a case the SB Core service will maintain a catalogue of the pre-allocated network slices and edge processes and of their configuration parameters, and it will perform an admission control for slice intent requests. If the slice intent requirements fit such pre-allocated resources it will assign and reserve them (or a part of them in case of shared, not isolated, slice requirements) to the requesting nApp.
- *Configurable Domains*: this case corresponds to the one where the programmability is handled by a black-box system external to the 5G-INDUCE framework. Upon any message requesting a lifecycle change for a slice, the SB Core service will ask the feasibility and the following (possible) materialization of this action to the external black-box system. The 5G-INDUCE Consortium decided to use the ETSI NFV-SOL 005 specification [8] as reference API to be used in this context: the onboarding of network service descriptors will be interpreted as feasibility of the slice-intent, while the network service instantiation command will be used to trigger the slice materialization. Configuration updates (if possible) will be requested by updating the network service.
- *Domains programmable at PaaS virtualization level*: This type of domains corresponds to environments where a full Kubernetes cluster is provided as a programmable resource. In such a case, the SB-OSS will activate the NFVCL services, but constraining them to the lifecycle management of cloud-native only network services. The same Kubernetes cluster will be used to realize namespaces to be used by the NAO to deploy nApp components.
- *Domains programmable at IaaS virtualization level*: It corresponds to cases where the SB-OSS has non-privileged access to IaaS environments (e.g., OpenStack). Similarly to the previous case, the NFVCL framework will be enabled to build and manage IaaS (and PaaS[1]) network services. Given the non-privileged access, the NFVCL will not have the possibility to manage and terraform some VIM-level resources, like for instance the dynamic creation of OpenStack external networks or Rule-Based Access Control (RBAC) among different projects (/tenants).

[1] 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.

- *Domains programmable at VIM level*: this case is very similar to the previous one, but with privileged access to the IaaS system. This will provide the NFVCL with the additional capacities of completely managing external networks and RBAC for resources shared among different projects.
- *Domains programmable at hypervisor Operating System level*: in this further case, both the NFVCL and the MetalCL will be enabled. The NFVCL will have full capabilities, while the MetalCL will have privileged access only to the Operating System of the hypervisors. 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*: in this case, the 5G-INDUCE system can exploit the maximum possible programmability levels. Both the NFVCL and the MetalCL will be enabled without constraints. The MetalCL will allow to 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 according to the 5G-INDUCE needs.

In any of the above introduced cases, upon the reception of a slice intent, the SB-OSS will check for the 5G slice type (i.e., eMBB, URLLC, and mMTC) and isolation requirements against catalogue resources/black-box system/directly manageable gNodeBs/Remote Radio Heads (RRHs) in order to verify the support for the requested spectrum in the coverage area and the availability of enough resources to meet the requirements.

4.2.2 The NFVCL Service

The objective of NFVCL is to provide a level of abstraction for the flexible and high-level management of the complete lifecycle orchestration of network services, VNFs and Physical Network Functions (PNFs) instantiated in the 5G infrastructure. More precisely, the NFVCL is able to build and to 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". Since the early MATILDA version, 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).

In a nutshell, the 5G-INDUCE NFVCL is in charge, upon request from the OSS-SB Core, to select the most suitable blueprints to be instantiated or the blueprint instances to be updated/reconfigured to satisfy the performance and functional requirements included in the slice request, and then to provide feedbacks to the OSS-SB Core when all the NFV services in the blueprint are ready and all the relevant VNFs and PNFs have been successfully configured. The NFVCL is also in charge of producing configuration commands and files to be passed to the network functions through Day 2 operations towards their VNF managers, and it is able to retrieve operation output, logs and performance metrics from them.

The reason why the NFVCL layer was introduced in the architecture is twofold: on the one hand, it was designed to support different NFVOs without the need to intervene on the OSS structure (since it implements standard ETSI NFV-SOL 004 [9], 005 [8] and 006 [2] APIs); on the other hand, it allows to cope with some shortcomings of ETSI NFV specifications, as explained herein. The Network Service Descriptors (NSDs) implemented in ETSI NFV are static, as the number of VNFs and the related internal/external connections are pre-determined in accordance with the VNF Descriptors (VNFD). Aside from the initial decision on the VIM to allocate the service on, each NSD dictates a fixed list of PNFs and SBA functions, including their connections, which makes it impossible, for example, to perform network service internal topology change

on any of the functions without destroying and re-deploying the whole service, let alone to modify the in-life operations on-the-fly upon changes on the related slice.

4.2.3 The MetalCL Service

The MetalCL is devoted to the management and terraforming of the VIMs, operating systems and bare-metal resources available in a testbed. In the presence of this service, physical servers and hardware network equipment, as well as their operating systems, can be dynamically managed on demand, in a similar way (although, of course, on different time scales[2]) as nApps and NFV services.

As shown in , the MetalCL connects with three external blocks that can be selectively enabled depending on the programmability level of the administrative infrastructure domain: the 5G-INDUCE NetDriver, the MaaS server, and the Ansible engine. The 5G-INDUCE NetDriver is a simple utility developed within this project in Python language. It uses the LLDP protocol to perform an automatic discovery of the physical topology, and it is meant to access the command line interface or to use the REST interface of physical networking devices in the computing facility, e.g., interconnection layer-2 managed switches (with VLAN or OpenFlow support), routers (optionally with the support of virtual routing functions) and firewalls. The NetDriver is used to create proper overlay networks and to manage the interconnectivity among servers as needed (i.e., to create proper networks for data- and control-plane for hosting OpenStack, Kubernetes, etc.).

NetDriver can also properly connect to servers and/or routing function to further physical devices connected to the local datacenters (e.g., base-stations, etc.)

The MaaS server is an open-source project developed by Canonical [3]. It allows discover, commission, deploy, and dynamically reconfigure a large network of individual bare-metal servers. The only prerequisites for servers in order to be used with MaaS are the presence of an IPMI-like (Intelligent Platform Management Interface [10]) system and the support of network boot operations through the PXE standard [11]. MaaS is able to catalogue bare-metal servers in a datacenter, to control their power states, and to install (on demand and as-a-Service) almost any operating system by properly configuring and administering the network(s) (mainly at the IP layer, while layer 2 interconnectivity and additional functions like gateways and firewalls are provided by the NetDriver service). MaaS exposes a complete set of REST APIs, which are consumed by the MetalCL to trigger any changes at the bare-metal level.

[2] The times to terraform bare-metal resources are generally much longer than the ones in the other layer of the architecture, since terraforming of bare-metal resources involves multiple reboot of hardware servers, which generally takes around 10 minutes per time.

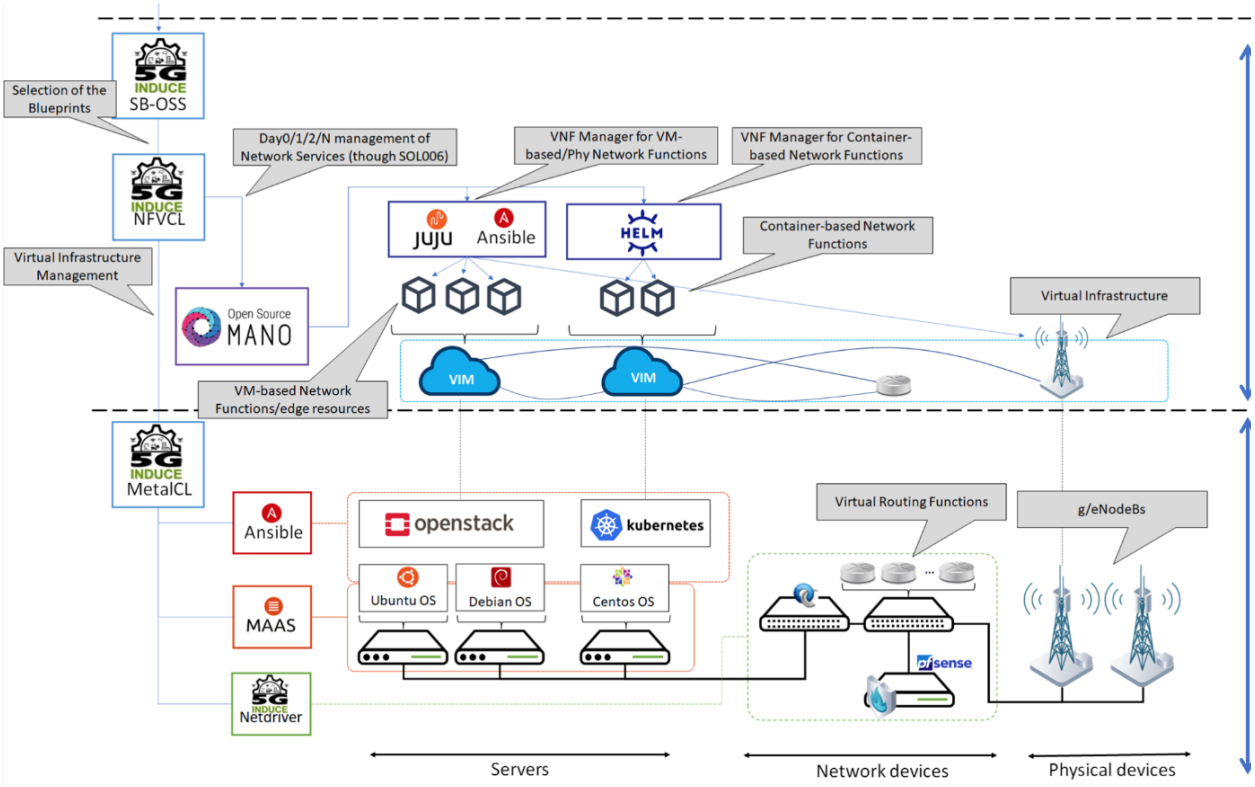

Figure 4: A deeper outlook on the SB-OSS.

Finally, MetalCL makes use of the Ansible engine to drive any software installation, application and OS reconfigurations over the bare-metal servers installed by MaaS. This engine is the one that provides the MetalCL with the capability of installing software dependencies and installing and managing the lifecycle, in a zero-touch fashion, of complex and distributed software like OpenStack or Kubernetes over one or more servers. MetalCL will provide also the possibility of configuring which server(s) to dedicate to a certain role in the complex applications (e.g., how many controller node servers in Kubernetes, how many compute, controller, or network nodes in OpenStack).

Moreover, also outlines how the NFVCL and the MetalCL provide the SB-OSS with the (adaptive) capability of controlling programmability and of managing the lifecycle of any artifact at the virtual and physical layers of the infrastructure. The junction point between the NFVCL and the MetalCL consists in the virtual infrastructure topology (including the set of VIMs, of virtual links, and of physical network functions like routers or base-stations), which can be created and modified as needed by the MetalCL. When the MetalCL is disabled, NFVCL obviously works over a static virtual infrastructure.

The NFVCL uses the virtual infrastructures to create the 5G network platform through the management of proper blueprints, NFV services, edge computing resources, and physical, VM-based, container-based network functions, which are updated, created and deleted according to the slice requests arriving from the NAO(s) managing the hosted NApps.

5 Integration

5.1 Within the Platform: NAO-OSS Interface

The NAO-OSS interworking interface is realized via a set of REST APIs (Representational State Transfer – Application Programming Interfaces) that rely on HTTP (Hypertext Transfer Protocol) methods.

Figure 5 provides an overview of the transactions that occur between the NAO and the OSS, which include:

- the request (from NAO towards OSS) for the creation of new 5G slices and the deletion of already deployed ones;
- the retrieval (by NAO) of the computing resources made available by the OSS;
- the request (from NAO towards OSS) for scaling deployed slices;
- the request (from NAO towards OSS) for scaling the computing resources that support nApps, to allow the relocation of nApp components.

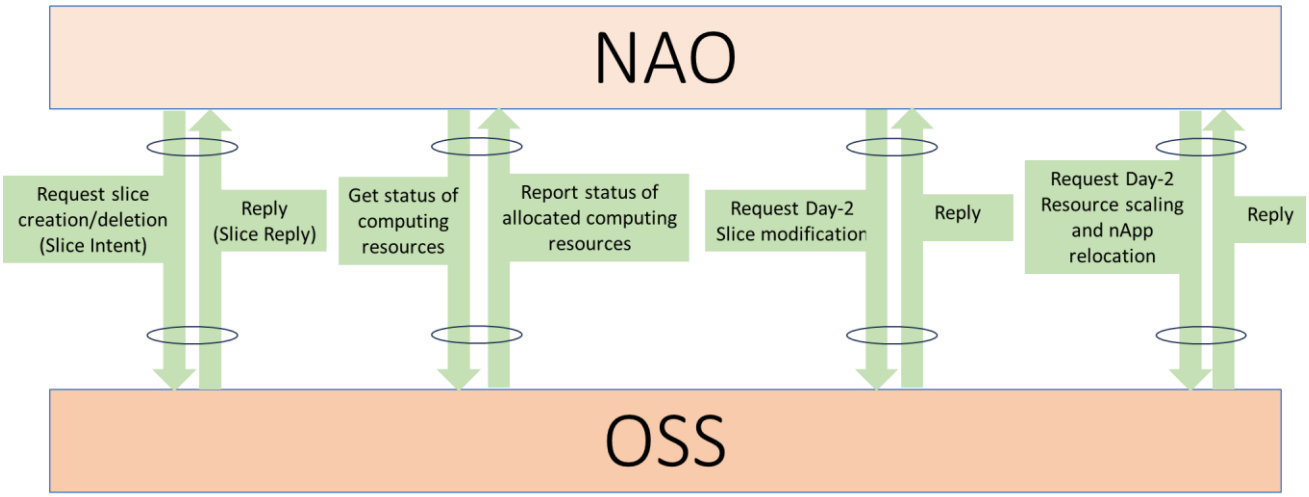

Figure 5: NAO-OSS Transactions.

The description of the information the NAO and the OSS exchange during the above listed transactions is included in the following.

5.1.1 Slice Creation

For the creation of new slices to support a nApp, the NAO posts to the OSS a message whose body, named Slice Intent, includes the description of the computing and memory resources as well as of communication services that the OSS needs to make available.

The key information included in the Slice Intent encompass:

- a unique identifier the NAO assigns to the Slice Intent, which the NAO and OSS will also share as identifier of the nApp deployment (if the creation request is successfully accomplished);
- the description of the nApp graph, expressed in terms of nApp component software modules and communication endpoints; these latter represent the abstract communication interfaces between either a couple of nApp components (“core endpoints”) or between one component and UEs via the 5G RAN (“edge endpoints”); both nApp components and endpoints have their own unique identifiers;

- the information of the location constraints for the deployment of nApp software components; to each nApp component is associated the 5G infrastructure PoP (i.e., data centre) that is appointed to host it;
- the dimensioning of computing and memory resources that each nApp component requires;
- the description of the communication services the nApp supporting 5G slices needs to provide;
- a set of 5G slices may be requested to be instantiated to support an nApp; each slice is described in terms of 5G PoPs it has to interconnect; furthermore, the communication services it provides are characterized in accordance to the 5G slice templates and QoS models defined in 3GPP TS 23.501 [12]; in particular, for each slice, the supported PDU Sessions and embedded QoS flows are dimensioned and each flow is associated to an nApp "edge endpoint";
- furthermore, a slice descriptor includes a list of UEs that are authorized to access the services that the slice provides; UEs are identified by UE's SIM cards' Integrated Circuit Card Identifier (ICCID) parameter.

The OSS, upon receiving and completing the processing of the Slice Intent, posts back to the NAO a reply, whose body, named Slice Reply, includes the following information:

- the outcomes of the OSS elaboration, i.e. whether the slice creation has been successfully completed or has failed;
- in case of a successful Slice Creation, the information the NAO needs to connect to and operate the Kubernetes cluster that the OSS set up;
- such information includes the Kubernetes namespace that the OSS has reserved for the NAO to deploy the nApp components listed in the specific Slice Intent, the Kubernetes controller API server URL and the certificates that the NAO has to utilize to authenticate towards the Kubernetes API Server;

in case of failure in the Slice Creation procedure, the list of exceptions that describe the causes for the failure.

5.1.2 Slice Deletion

The NAO can request the OSS to delete an instantiated slice and free the relevant virtual resources by executing a delete operation on the specific OSS endpoint that refers to the original Slice Intent identifier.

The OSS will reply to the NAO by posting a Slice Reply that reports the outcome of the operation, which may be successful or failed.

5.1.3 Retrieve Available Computing Resources

The NAO can get the list of locations (i.e., 5G infrastructure PoPs) and Kubernetes clusters that the OSS has reserved to the same NAO, following the set of preceding completed transactions.

The report the OSS provides to the NAO includes:

- the list of geographical areas the OSS makes available to provision Kubernetes clusters, which the NAO can utilize to deploy nApp components;
- information about the already provisioned Kubernetes clusters created within the listed geographical areas, together with the list of Kubernetes worker nodes available in the clusters.

The NAO may process the retrieved information to identify the 5G resources it has consumed and the ones that it may access to modify the deployment of nApps to, e.g., relocate nApp components (see the following description of Day-2 operations).

5.1.4 Day-2 Scaling of Slice Networking Resources

The NAO can request an update of the networking resources that deployed nApps utilize by patching the configuration data of existing slices, which are addressed by the unique identifiers associated to the Slice Intents shared between the NAO and the OSS. The body of the message that the NAO sends to the OSS

consists of a subset of the Slice Intent and includes, in particular, the updated descriptors of the PDU Sessions and embedded QoS flows associated to nApp "edge endpoints".

Following a request from the NAO, the OSS replies by posting to the NAO a message that includes the outcomes of the message processing operated by the OSS (i.e., whether the modifications could be successfully applied or not) and, in case of failure of the slice modification, a list of exceptions that describe the reasons for the failure.

5.1.5 Day-2 Scaling of Computing Resources

The NAO can request the OSS to allocate resources that the NAO needs to modify the deployment of nApp components.

To this purpose, the NAO can patch the list of 5G network PoPs by providing the OSS with a new list of location constraints, which describes new associations between a subset of nApp components (the ones that shall be relocated) and the 5G infrastructure PoPs. The OSS will reserve the additional computing/storage resources, located in the identified geographical areas, and assign them to the Kubernetes Cluster that already supports the nApp. Furthermore, the OSS will extend the deployed 5G slice to provide connectivity to the new locations.

After the OSS has successfully completed the allocation of the new computing nodes and updated the 5G slice, the NAO will redeploy the specified nApp components from the current geographical areas to the new ones.

The OSS signals the completion of the operation by posting to the NAO a reply message that reports whether the operation was successful or unsuccessful, and, in case of failure, the exceptions that describe the relevant probable causes.

5.2 With the Infrastructures: DevOps and ExFas

As described in Section 4, the 5G-INDUCE Platform is designed to manage administrative domains with heterogeneous levels of programmability, which are identified as follows:

- Fully programmable facilities, such as the DevOps testbed, which allow to be operated at bare metal level;
- Partially programmable Experimentation Facilities (ExFas), which embed their own proprietary (black-box) 5G management layer, and allow embedded virtual computing/storage resources to be managed at either IaaS or PaaS level;
- Non-programmable ExFas, which are fully manageable only by their own proprietary control-management plane, and where resources will be statically assigned to 5G-INDUCE.

From the perspective of the 5G-INDUCE platform, the project ExFas and the DevOps testbed correspond to different administrative domains, each of which is managed by a specifically featured instance of the SB-OSS.

Figure 6-9 depict the different architectural solutions for the integration of the 5G-INDUCE Platform with the DevOps Testbed and the various ExFas; a summary description of the different scenarios and of relevant features are described herein.

5.2.1 DevOps

The DevOps testbed, located in the CNIT premises in Genoa, Italy, 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 (e.g., as project environments) that can emulate complete 5G network environments, as well as to manage and configure their respective physical/virtual resources through a Metal-as-a-Service (MaaS) approach.

The testbed infrastructure is an integration of both general- and special-purpose equipment to realize underlying 5/6G networks substrates and relative slices, as well as edge computing resources. The hardware infrastructure currently included 35 servers for a total of 1300 cores, 10 TB of RAM and over 100 TB of storage, interconnected through 8 high-speed switches, along with GPUs, O-RAN radio units, access network emulators, and several testing and measurement tools including traffic generators, power meters, etc.

The testbed hosts the 5G-INDUCE platform, e.g., the NAO and the OSS, and its SB-OSS includes all of the modules defined within the project, namely, the South-Bound Core service, the NFVCL, and the MetalCL. As a result, it offers full programmability to the users even including the configuration of bare-metal resources and the dynamic creation, scaling, and terraforming of IaaS and PaaS environments.

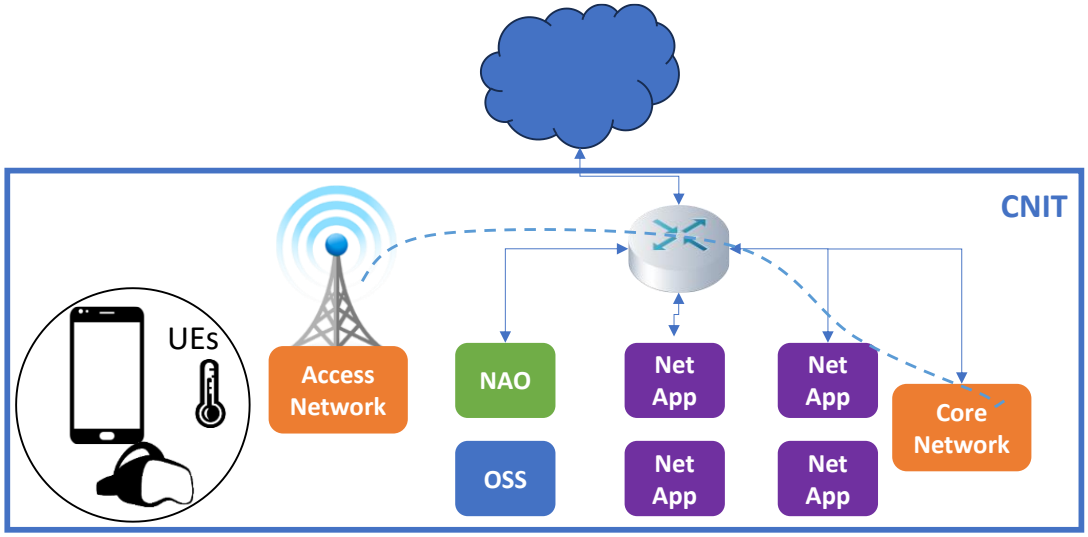

Figure 6: High level architecture of the DevOps testbed.

5.2.2 ExFa IT

The Platfrom integration within the ExFa has been obtained by interconnecting via a VPN the WindTre 5G Core Network with the DevOps hosting the 5G-INDUCE Platform. The WindTre 5G network provides the connectivity between the UEs and the nApps that are all deployed in a replica of the DevOps infrastructure hosted in the CNIT facility. Hence, 5G slices are statically provisioned within the WindTre network, while nApp components can be flexibly deployed, under the Platform control, over the DevOps replica.

WindTre ExFa falls under the category of “partially programmable facility”. As a matter of fact, the programmability is limited to the management of computing resources (thanks to the integration of the DevOps Testbed replica), while 5G services are statically provisioned and out of the control of the 5G-INDUCE Platform.

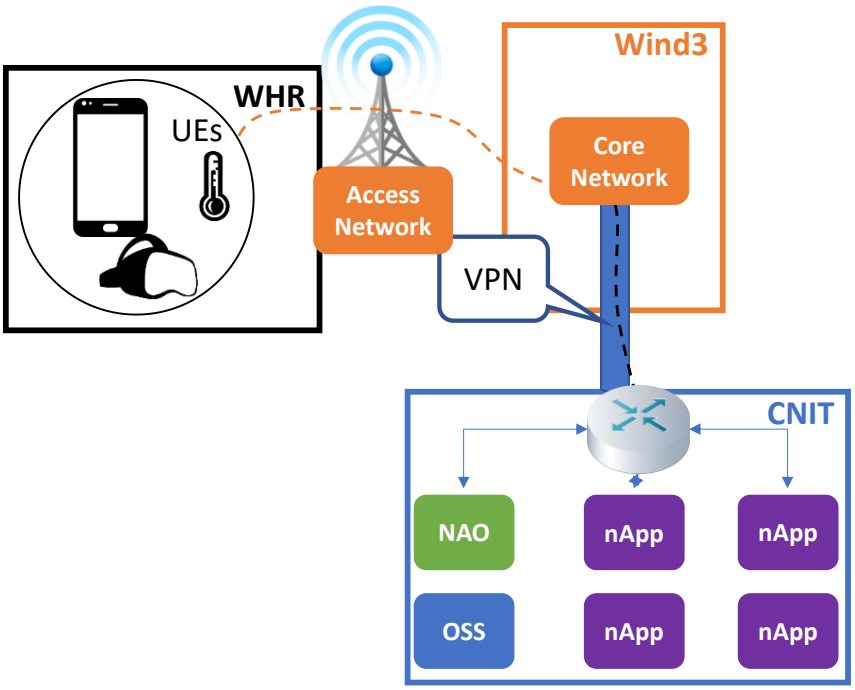

Figure 7: High level architecture of the ExFa-IT testbed.

5.2.3 ExFa-GR

The OTE ExFa embeds both edge computing facilities and 5G network components (RAN and Core).

The edge computing facility consists of a Kubernetes cluster the OSS and NAO can access to monitor the status of the resources and deploy nApp components.

The 5G-INDUCE Platform interworks with the OTE's 5G Core controller through interfaces provided by the Slice Manager module. The Slice Manager exports towards the OSS the catalogue of 5G slices that have been deployed via the OTE's 5G Core GUI. The catalogue is composed of two lists, one for the eMBB and one for the URLLC slice type. For both types, the slices expose the following information:

- sliceId: unique identifier of the slice;
- site: fixed, "Athens";
- expDataRateUL: upload data rate;
- expDataRateDL: download data rate;
- userDensity: maximum number of UEs that can be connected to the slice;
- userSpeed: speed of the UE traffic;
- trafficType: UDP or TCP;
- IMSI: list of the IMSIs connected to the slice.

When the OSS receives a slice intent request from the NAO, the NFVCL (that is located in the same server of the slice manager, see Figure 8) uses the expDataRateUL and DL, userDensity and userSpeed parameters to select the deployed slice that best matches nApp requirements. Then by using a specific API provided by the Slice Manager, the OSS associates the IMSIs the NAO listed in the Slice Intent to the selected slice, hence enabling the UEs to access the slice itself. At runtime, the expDataRateUL and expDataRateDL parameters can be modified if requested by the NAO.

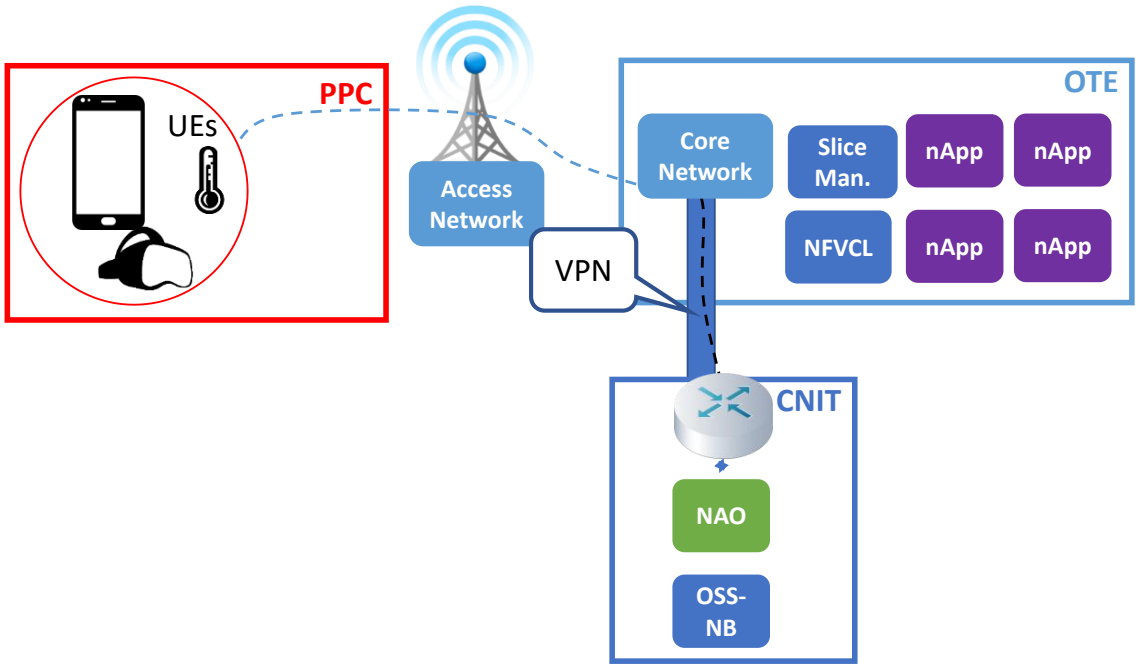

Figure 8: High level architecture of the ExFa-GR testbed.

The OTE ExFa is classifiable as a "partially programmable facility", in that it allows to select among a set of predefined 5G services the one more suitable to support the deployed nApp.

5.2.4 ExFa-SP

The ExFa falls into the "fully programmable" category, whilst the programmability affects only some 5G service parameters.

In fact, the ExFa provides the NFVCL with APIs that allow the deployment of a parametrizable slice. Configurable slice parameters include:

- site: fixed, "FORD_VALENCIA"
- coverageArea: can be "FORD_INDOOR" or "FORD_OUTDOOR" as the ExFa includes two gNodeBs that can be alternatively selected to be connected with the slice;
- uLThptPerSlice: uplink throughput
- dLThpPerSlice: downlink throughput.

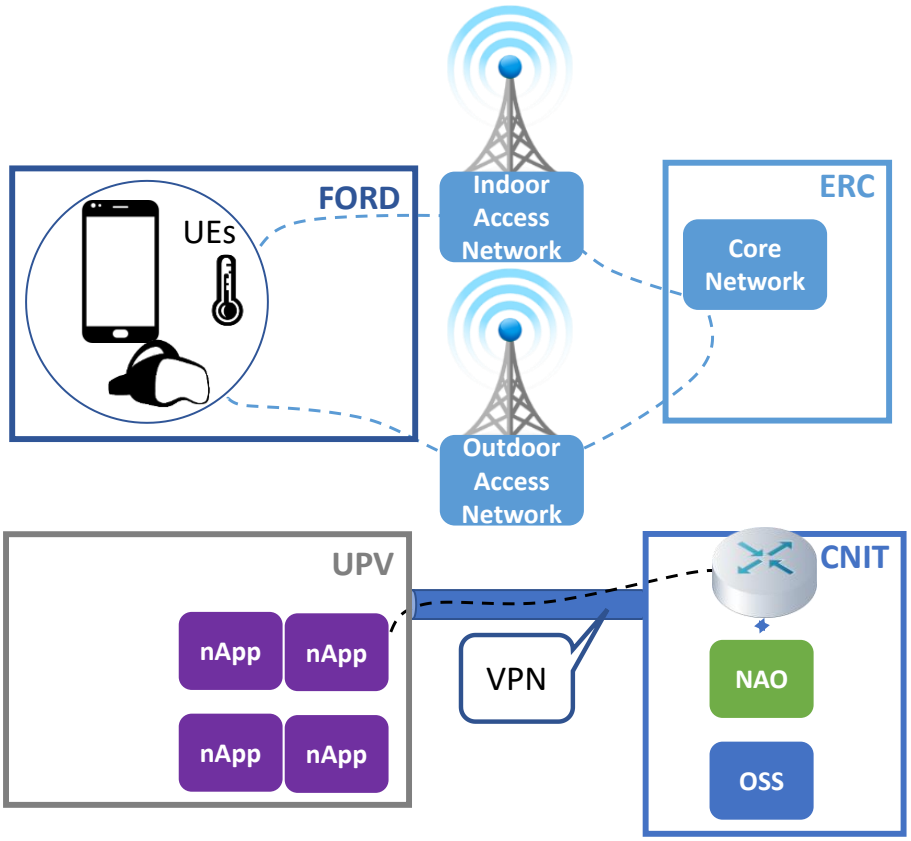

Figure 9: High level architecture of the ExFa-SP testbed.

6 Conclusions

This deliverable described the completed 5G-INDUCE Platform, including the NAO and the OSS, as well as the interworking interface among them.

The NAO is responsible for the nApp registration, deployment and runtime policies, providing application monitoring, negotiating slices and managing the operational state of the nApp within the scope of an allocated slice. The NAO interplays with the OSS, which is in charge of managing all functions and operations required for the nApp placement over edge computing facilities, for its connection to a network slice, and for maintaining the information on all the data regarding the deployed applications, network services, and available infrastructure resources.

The 5G-INDUCE Platform allows to manage administrative domains with heterogeneous levels of programmability, spanning from bare-metal, full programmability to non-programmable ExFas statically assigning resources to 5G-INDUCE.

References

[1] 5G-INDUCE Grant Agreement Number 101016941

[2] 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

[3] Canonical MaaS, URL: https://maas.io/

[4] https://medium.com/@karthik.jeyapal/system-design-patterns-producer-consumer-pattern-45edcb16d544.

[5] https://prometheus.io/docs/prometheus/latest/querying/basics/

[6] MongoDB, URL: www.mongodb.com/

[7] Prometheus, URL: https://prometheus.io/

[8] ETSI GS NFV-SOL 005, "Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point", V2.7.1, 2020

[9] ETSI GS NFV-SOL 004, "Network Functions Virtualisation (NFV) Release 3; Protocols and Data Models; VNF Package and PNFD Archive specification", V3.5.1, 2021

[10] Intel, "IPMI Specification", V2.0, Rev. 1.1,
URL: https://www.intel.it/content/www/it/it/products/docs/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html

[11] Intel, "Preboot Execution Environment (PXE) Specification", V2.1, 1999,
URL: https://web.archive.org/web/20131102003141/http://download.intel.com/design/archives/wfm/downloads/pxespec.pdf

[12] 3GPP TS 23.501 V17.7.0, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; System architecture for the 5G System (5GS); Stage 2 (Release 17)", December 2022.