



**5TH GENERATION END-TO-END NETWORK, EXPERIMENTATION,
SYSTEM INTEGRATION, AND SHOWCASING**

[H2020 - Grant Agreement No. 815178]

Deliverable D3.1

Management and Orchestration (Release A)

Editor J. Melián (ATOS)

Contributors ATOS (Atos Spain SA), UMA (Universidad de Malaga), NCSRD (National Center for Scientific Research “Demokritos”), FhG (Fraunhofer Gesellschaft Zur Foerderung der Angewandten Forschung E.V.), UPV (Universitat Politecnica de Valencia), UNIS (University of Surrey), LMI (L.M. Ericsson Limited), AVA (Avanti Hylas 2 Cyprus Limited), TID (Telefónica I+D), INF (INFOLYSIS P.C.)

Version 1.0

Date October 15th, 2019

Distribution PUBLIC (PU)



List of Authors

NCSRD	NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”
G.Xilouris, S. Kolometsos, T. Anagnostopoulos	
UMA	UNIVERSIDAD DE MÁLAGA
F. Luque, B. Valera, A. Díaz, P. Merino	
LMI	L.M. ERICSSON LIMITED
A.M. Cristina, E. Aumayr, J. McNamara	
ATOS	ATOS SPAIN SA
E. Jimeno, J. Melian, S. Castro	
FhG	FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V.
M. Emmelmann, F. Eichhorn, T. Briedigkeit, S.K. Rajaguru, A. Prakash	
UPV	UNIVERSITAT POLITECNICA DE VALENCIA
J. Suárez de Puga	
UNIS	UNIVERSITY OF SURREY
S. Vahid, Y. Rahulan	
AVA	AVANTI HYLAS 2 CYPRUS LIMITED
A. Perentos, S. Watts	
SHC	SPACE HELLAS (CYPRUS) LTD
D. Lioprasitis, G. Gardikis	
INT	INTEL DEUTSCHLAND GMBH
V. Frasca	
INF	INFOLYSIS P.C.
C. Sakkas, A. Papaioannou	
TID	TELEFONICA INVESTIGACION Y DESARROLLO SA
A. Flórez, D. Artuñedo	

Disclaimer

The information, documentation and figures available in this deliverable are written by the 5GENESIS Consortium partners under EC co-financing (project H2020-ICT-815178) and do not necessarily reflect the view of the European Commission.

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The reader uses the information at his/her sole risk and liability.

Copyright

Copyright © 2019 the 5GENESIS Consortium. All rights reserved.

The 5GENESIS Consortium consists of:

NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”	Greece
AIRBUS DS SLC	France
ATHONET SRL	Italy
ATOS SPAIN SA	Spain
AVANTI HYLAS 2 CYPRUS LIMITED	Cyprus
AYUNTAMIENTO DE MALAGA	Spain
COSMOTE KINITES TILEPIKOINONIES AE	Greece
EURECOM	France
FOGUS INNOVATIONS & SERVICES P.C.	Greece
FON TECHNOLOGY SL	Spain
FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V.	Germany
IHP GMBH – INNOVATIONS FOR HIGH PERFORMANCE MICROELECTRONICS/LEIBNIZ-INSTITUT FUER INNOVATIVE MIKROELEKTRONIK	Germany
INFOLYSIS P.C.	Greece
INSTITUTO DE TELECOMUNICACOES	Portugal
INTEL DEUTSCHLAND GMBH	Germany
KARLSTADS UNIVERSITET	Sweden
L.M. ERICSSON LIMITED	Ireland
MARAN (UK) LIMITED	UK
MUNICIPALITY OF EGALEO	Greece
NEMERGENT SOLUTIONS S.L.	Spain
ONEACCESS	France
PRIMETEL PLC	Cyprus
RUNEL NGMT LTD	Israel
SIMULA RESEARCH LABORATORY AS	Norway
SPACE HELLAS (CYPRUS) LTD	Cyprus
TELEFONICA INVESTIGACION Y DESARROLLO SA	Spain
UNIVERSIDAD DE MALAGA	Spain
UNIVERSITAT POLITECNICA DE VALENCIA	Spain
UNIVERSITY OF SURREY	UK

This document may not be copied, reproduced or modified in whole or in part for any purpose without written permission from the 5GENESIS Consortium. In addition to such written permission to copy, reproduce or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

Version History

Rev. N	Description	Author	Date
1.0	Release of D3.1	ATOS	15 th - October - 2019

LIST OF ACRONYMS

Acronym	Meaning
5G PPP	5G Infrastructure Public Private Partnership
5G-IA	The 5G Infrastructure Public Private Partnership
AP	Access point
API	Application Programmable Interface
AR	Augmented Reality
BYOD	Bring Your Own Device
CA	Carrier Aggregation
CESC	Cloud-Enabled Small Cell
CO	Central Office
CP	Connection Point
CoMP	Coordinated Multi-Point transmission/reception
CPRI	Common Public Radio Interface
C-RAN	Cloud-RAN
CSP	Content Service Provider
CUPS	Control and User Plane Separation
DoS	Denial of Service
DDoS	Distributed Denial of Service
DU	Digital Unit
eICIC	Enhanced Inter-Cell Interference Coordination
eMBB	Enhanced Mobile Broadband-5G Generic Service
eMBMS	Evolved Multimedia Broadcast Multicast Services
EMS	Element Management System
eNB	eNodeB, evolved NodeB, LTE eq. of base station
ETSI	European Telecommunications Standards institute
EU	European Union
EPC	Evolved Packet Core
EUTRAN	Evolved Universal Terrestrial Access network
FDD	Frequency Division Duplexing
gNB	gNodeB, 5G NR, next generation NR eq. of base station
GPP	General Purpose Processor
G-VNFM	Generic VNF Manager
HetNet	Heterogeneous Network
H-RAN	Heterogeneous RAN
ICIC	Inter-Cell Interference Coordination
ICMP	Internet Control Message protocol
IDS	Intrusion Detection System
IOT	Internet of Things
ISG	Industry Standardization Group
KPI	Key Performance Indicator
KVM	Kernel-based Virtual Machine
LPWA	Low Power Wide Area

Acronym	Meaning
LTE	Long-Term Evolution
LTE-A	Long-Term Evolution - Advanced
MANO	NFV MANagement and Organisation
MCS	Mission Critical Services
MEC	Mobile Edge Computing
MIMO	Multiple Input Multiple Output
MME	Mobility Management Entity
mMTC	Massive Machine Type Communications-5G Generic Service
MONROE	Measuring Mobile Broadband Networks in Europe.
MPTCP	Multipath TCP
NFV	Network Function Virtualisation
NFVI	Network Function Virtualisation Infrastructure
NFVO	NFV Orchestrator
NMS	Network Management System
NR	New Radio
NS	Network Service
NSMF	Network Slice Management Function
NST	Network Slice Template
OAI	Open Air Interface
OAM	Operations, Administration & Management
ODL	OpenDayLight
OF	OpenFlow
ONAP	Open networking Automation Platform
OPNFV	Open Platform for NFV
ORI	Open Radio Interface
OSM	Open Source MANO
OTT	Over-The-Top
PCell	Primary Cell
PCI	Physical Cell ID
PCRF	Policy and Charging Rules Function
PDCP	Packet Data Convergence Protocol (PDCP)
PoP	Point of Presence
P-GW	Packet Data Node Gateway
PNF	Physical Network Functions
PPDR	Public Protection and Disaster Relief Systems
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RRH	Remote Radio Head
RRM	Radio Resource management
RU	Radio Unit
SDK	Service Development Kit
SDN	Software Defined Network
SDR	Software Defined Radio

Acronym	Meaning
SeCaaS	SeCurity as a Service
SLA	Service-Level Agreement
SP	Service Platform
STA	Station
TCP	Transmission Control Protocol
UAV	Unmanned Aerial Vehicles
UDP	User datagram Protocol
UE	User Equipment
UI	User Interface
uRLLC	Ultra-Reliable, Low-Latency Communications
VDU	Virtual Deployment Unit
vEPC	Virtual EPC
VIM	Virtual Infrastructure Manager
VL	Virtual Link
VLAN	Virtual Local Area Network
VNF	Virtual Network Function
VNFM	VNF Manager
VnV	Validation and Verification
WIM	WAN Infrastructure Manager
WSMP	Wifi Service Management Platform

Executive Summary

The aim of the present document is to describe a midterm version of the design and implementation of the Management and Orchestration (MANO) component. As shown in Figure 1 the MANO component is part of the Management and Orchestration layer of the 5GENESIS architecture.

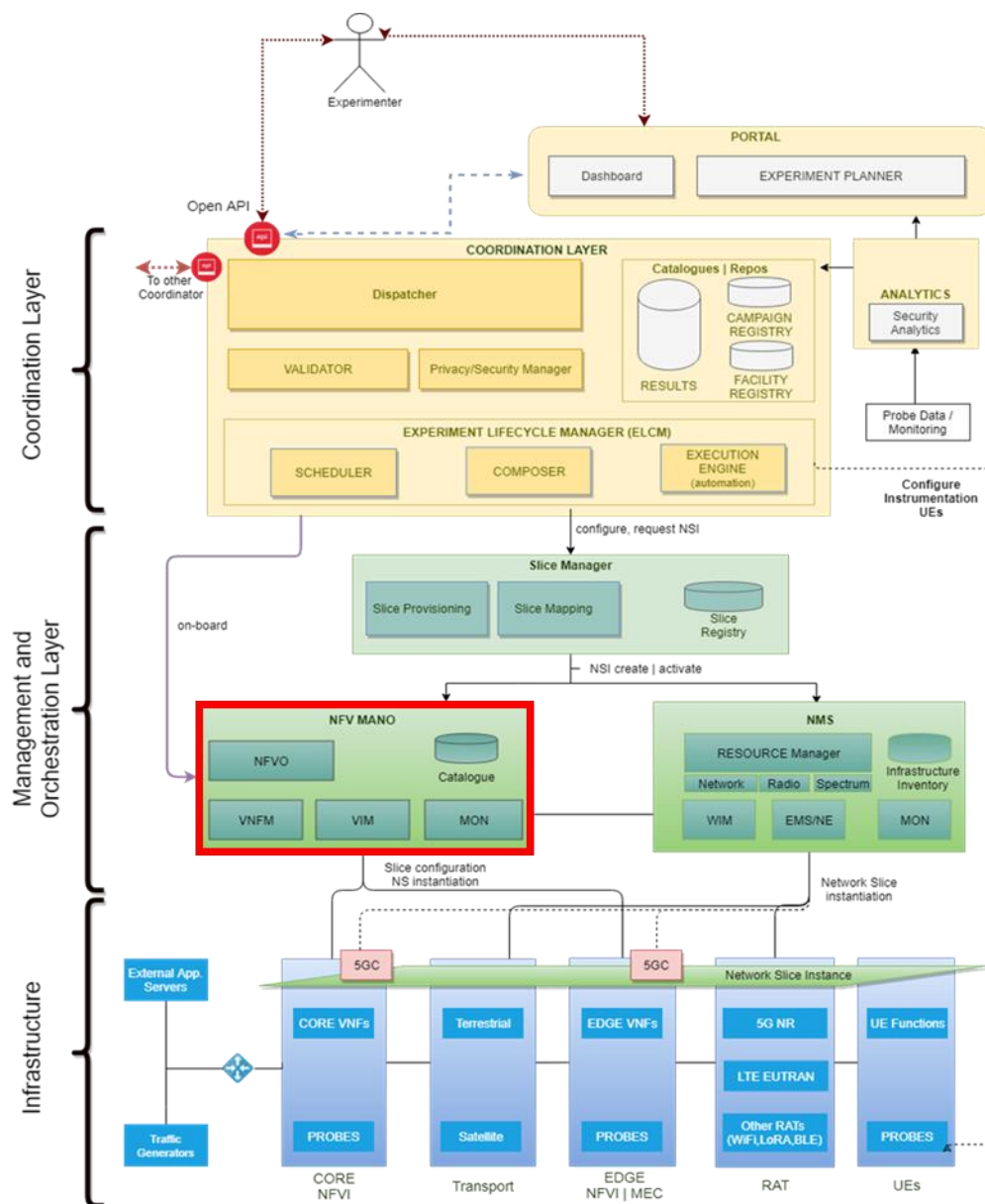


Figure 1 MANO component in the 5GENESIS architecture

This document begins studying the state of the art of different orchestrators (NFVO), the most used and popular in the sector, valuing benefits and problems of using each one of them to check their adaptability to the 5GENESIS needs. The same occurs with the Virtual Infrastructure Managers (VIM), that are brought to study in in this deliverable.

After defining the common architecture for the Management and Orchestration for 5GENESIS, and also the interactions with the neighbour components, both, the NFVO and the VIM, need to fulfil certain requirements gathered and fully described to be 5GENESIS compliant.

Additionally, this document will provide an exhaustive description of how this solution is being implemented per platform, as well as the current status of the implementation in the platform and finally a roadmap for a fully functional MANO.

This deliverable will serve as a basis for an updated and final version of the same document D3.2 Management and orchestration (Release B) at M30.

Table of Contents

LIST OF ACRONYMS	6
1. INTRODUCTION	14
1.1. Purpose of the document	14
1.1.1. Document dependencies	14
1.2. Structure of the document	14
1.3. Target audience	15
2. STATE OF THE ART	16
2.1. NFVO- Orchestration	16
2.1.1. Open Source MANO (OSM)	17
2.1.2. Open Baton.....	18
2.1.3. Open Networking Automation Platform (ONAP)	19
2.1.4. Cloudify.....	20
2.1.5. SONATA.....	22
2.2. Virtual Infrastructure Manager (VIM)	23
2.2.1. OpenStack.....	23
2.2.2. OpenNebula.....	24
2.2.3. Kubernetes.....	24
3. MANO SOLUTION IN 5GENESIS	25
3.1. System specifications.....	25
3.1.1. NFV MANO (Network Function Virtualisation Management and Orchestration)...	25
3.1.2. NMS (Network Managemetn System)	26
3.1.3. MANO Requirements.....	26
3.2. MANO architecture.....	27
3.2.1. NFV Orchestrator (NFVO)	28
3.2.2. VNF Manager (VNFM).....	28
3.2.1. Virtualisation Infrastructure Manager (VIM)	28
3.2.2. Monitoring.....	29
3.2.3. Catalogues	29
3.3. Role in 5GENESIS architecture	31
3.3.1. Interfaces.....	33
High level description of the interfaces	33
3.3.2. Roadmap.....	34

4. RELEASE A SUMMARY AND FUTURE PLANS.....	36
4.1. Malaga platform	36
4.1.1. MANO solution adopted and features.....	36
4.1.2. Platform status (Release A).....	37
4.1.3. Future plans.....	38
4.2. Berlin platform.....	39
4.2.1. MANO solution adopted and features.....	39
4.2.2. Extensions developed as part of the Project	39
4.2.3. Platform status (Release A).....	40
4.2.4. Future plans	42
4.3. Athens platform.....	43
4.3.1. MANO solution adopted and features.....	43
4.3.2. Extensions developed as part of the Project	44
• MNL WIM	44
• MNL EMS.....	44
4.3.3. Platform status (Release A).....	45
4.3.4. Future plan	45
4.4. Limassol platform	46
4.4.1. MANO solution adopted and features.....	46
4.4.2. Extensions developed as part of the Project	48
4.4.3. Platform status (Release A).....	48
4.4.4. Future plans.....	49
4.5. Surrey platform.....	50
4.5.1. MANO solution adopted and features.....	50
4.5.2. Policy Engine (APEX)	51
4.5.3. Platform status (Release A).....	55
4.5.4. Future plans	55
5. CONCLUSIONS	57
REFERENCES.....	58
ANNEX 1 – VNF DESCRIPTORS	60
ANNEX 2 – NS DESCRIPTORS	62
ANNEX 3 – ONAP POLICY ARCHITECTURE	64
• Policy Creation.....	66
• Policy Distribution	67

- Policy Decision and Enforcement 67

1. INTRODUCTION

1.1. Purpose of the document

This deliverable presents the work done for the 5GENESIS ecosystem under the scope of the task 3.1 within WP3, focusing on the Management and Orchestration (MANO) of the 5GENESIS platforms.

The outcome will be the description of a set of components, located in the MANO Layer, that will be common to all the platforms: Athens, Berlin, Limassol, Malaga and Surrey. Even when the implementation and the election of the subcomponents is free, based on the technologies used in each platform, the input, output and behaviour will be the same.

1.1.1. Document dependencies

This document, as most of the WP3 deliverables, is based on specifications, requirements and assumptions as discussed in the first release of the Architecture related deliverables. The table below summarizes the relevance towards the deliverables produced by WP2.

Table 1 - Document dependencies

id	Document title	Relevance
D2.1 [29]	Requirements of the Platform	The document sets the ground for the first set of requirements related to supported features at the testbed for the facilitation of the Use Cases.
D2.2 [26]	5GENESIS Overall Platform Design and Specifications	The 5GENESIS platform architecture is defined in this document. The list of functional components to be deployed in each testbed is defined.

1.2. Structure of the document

The document is organized as follows:

- The acronym table provides an overview of the terminology used within this deliverable.
- Section 1 describes the introduction and the scope of the document.
- Section 2 provides comprehensive review of the state of the art of different orchestrators.
- Section 3 describes an overview of the MANO architecture and how it is implemented and adapted in 5GENESIS. In this section the requirements are presented and also will be defined the user specifications and the technical requirements of the solution.
- Section 4 presents the MANO approach per platform, including which solution has been adopted along with its features, the platform status and the roadmap.

- Finally, section 5 contains conclusions and summarizes the deliverable with the future steps to follow.

1.3. Target audience

As the MANO layer is a common component to all the platforms, this deliverable is addressed to the Project Consortium to (i) have a wider understanding of the technologies available to carry out this task, (ii) create a common approach to the implementation, configuration and usage of the components that conform the MANO layer, (iii) be aware of the different technologies used in the different platforms to perform the same tasks and how each one implements the same blueprint, and finally, (iv) be able to map further work, reckoning the labour done in other platforms having the same roadmap as a base.

Additionally, general public may also get a better understanding of what the Management and Orchestration does and how it facilitates the work for the automation of the Experiment deployments, the different approaches of the implementation taken by each platform and the current status of such implementation.

2. STATE OF THE ART

Network Functions Virtualisation (NFV) technology focuses on addressing problems of the new telecommunications networks. The fundamental framework called NFV Management and Network Orchestration (NFV MANO) is presented in order to provide a vision of the architecture. The NFV Orchestrator (NFVO) and the VNF Manager (VNFM) are functional blocks especially important within this framework, explained in detail in section 3.2. Important concepts such as Virtual Network Function (VNF) and VNF/Network Service Descriptors (VNFD/NSD) are also important in this scope [2].

Virtualisation promises several benefits to network operators, including [3]:

- Reduce costs in purchasing network equipment via migration to software on standard servers.
- Efficiencies in space, power, and cooling.
- Faster deployment time.
- Flexibility – elastic scale up and scale down of capacity.
- Access to a large independent software community, including open source.

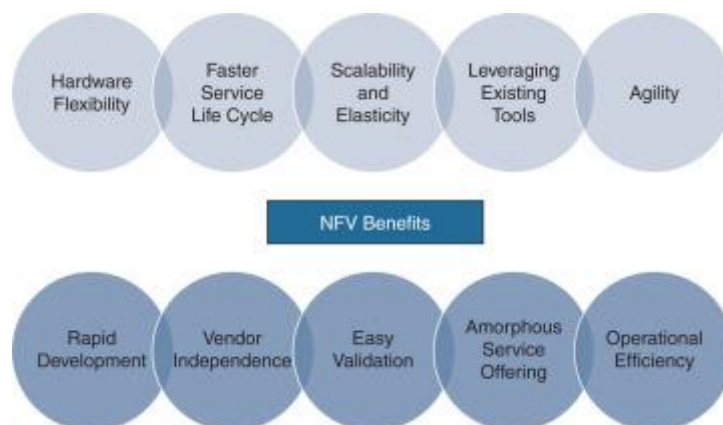


Figure 2 - Benefits of NFV [1]

Following sections investigate the open source solutions for network virtualization technology, in order to have a better overview of the NFV projects following the ETSI NFV information models [4]. Research on State of the art in the MANO layer will focus mainly in the NFVO and VIM technologies.

2.1. NFVO- Orchestration

MANO solutions, mainly composed by NFVO (Network Function Virtualization Orchestrator) and VNFM (VNF Manager) entities, are introduced to address the management and orchestration functionality in the NFV MANO framework. NFVO is a key component involved in resource orchestration and network service orchestration, to provide a network service managing the compute, storage and network resources available in the Infrastructure to provide a full NFVI (NFV Infrastructure). NFVO provides governance for the VNF instances that compose the virtual services across different points of presence (PoP) in the infrastructure.

Moreover, NFVO is used to automate and optimize the use of resources and comply with the type of ITU service. Some of the most important solutions existing nowadays are presented below.

2.1.1. Open Source MANO (OSM)

As already introduced in D2.2 Appendix [5], OSM [6] is an ETSI-hosted project, in constant and rapid evolution, that is developing an open source NFV MANO platform aligned with ETSI NFV MANO reference architecture [7], respecting appropriate IFA working group specifications, and meeting the requirements of production for NFV networks.

The OSM architecture has a clear split of orchestration functions between resource and service orchestrators. Since its third release, it integrates open source software initiatives such as Riptide [8] as network service orchestrator and GUI, Open MANO as resource orchestrator (NFVO), and Juju charms [9] as configuration manager tool (G-VNFM). The resource orchestrator supports both cloud and SDN environments. The service orchestrator can provide VNF and NS lifecycle management and consumes open information and/or data models, such as YANG. MANO architecture covers only single administrative domain.

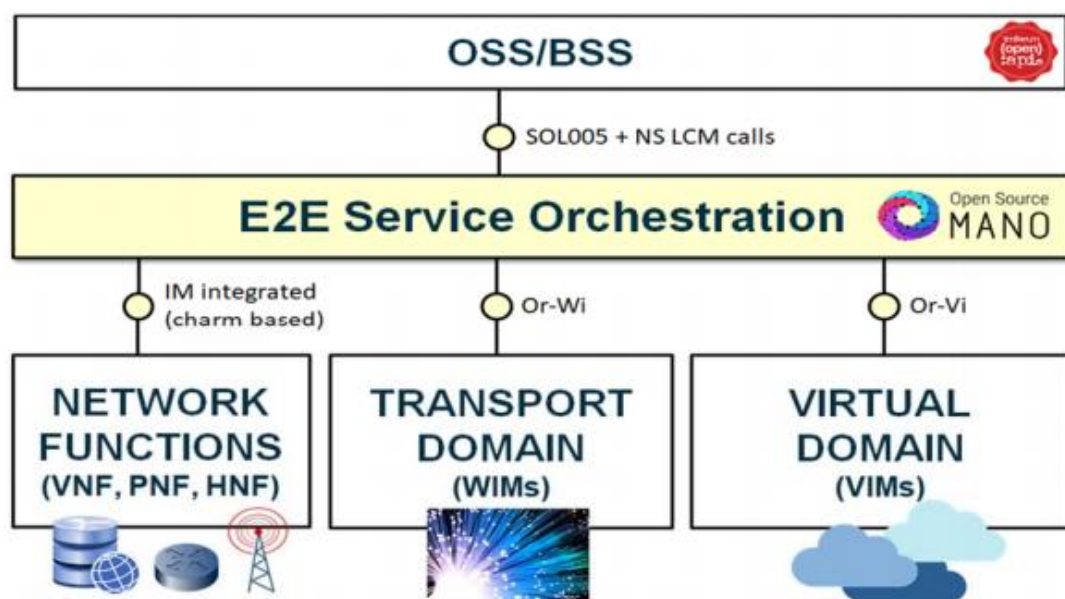


Figure 3 - Cross-domain End-to-End Service Orchestration with OSM [11]

Key reference points, such as Or-Vnfm and Or-Vi interfaces might be identified within OSM components. The VNF and Network Service (NS) catalogue are explicitly present in an OSM service orchestrator (SO) component.

The project recently launched its Release Six [10] in June 2019, including new set of capabilities to provide end-to-end orchestration across heterogeneous networks and cloud technologies [12]:

- Network Service primitives and the extension of its Service Assurance (SA) framework, which now can control, store and react to a much wider set of events and conditions in the context of running Network Services and Slices, easing the management of complex services.

- Enlargement of underlying technologies that are supported by OSM. New connectors have been developed and improved for FOG05 Edge clouds, TAPI-based transport networks, VMware cloud, and public clouds. Similarly, additional features have been added to support additional EPA attributes and ameliorate the support of underlays with the addition of multi-segment networks.
- Operational effectiveness at the edge. Critical to a successful 5G strategy and emerging business models for edge-based compute, connecting edge and core, to provide reusable services that span the full telco topology and enable both 5G infrastructure and third-party app ecosystems for the edge in VMs and containers.

To these improvements, we have to add features from previous releases like [11]:

- Network slicing for 5G.
- Support for physical and hybrid network functions.
- Multi-domain orchestration providing dynamic multi-site inter-datacentre connectivity.
- Monitoring and policy framework providing closed-loop operations.
- Capabilities for VNF configuration and deployment.

All these enhancements lead to a more flexible operator's experience, with an improved control over orchestration roles (fine-grained control of operations per role and project), and better real-time feedback to the operator, along with various improvements to ease VNF onboarding and testing phases.

2.1.2. Open Baton

As already described in D2.2 [5], Open Baton [13] is an open source reference implementation of the NFVO based on the ETSI NFV MANO specification [14] and the TOSCA standard [15]. It comprises a vendor-independent platform (i.e. interoperable with different vendor solutions), which is easily extensible for supporting new functionalities and existing platforms.

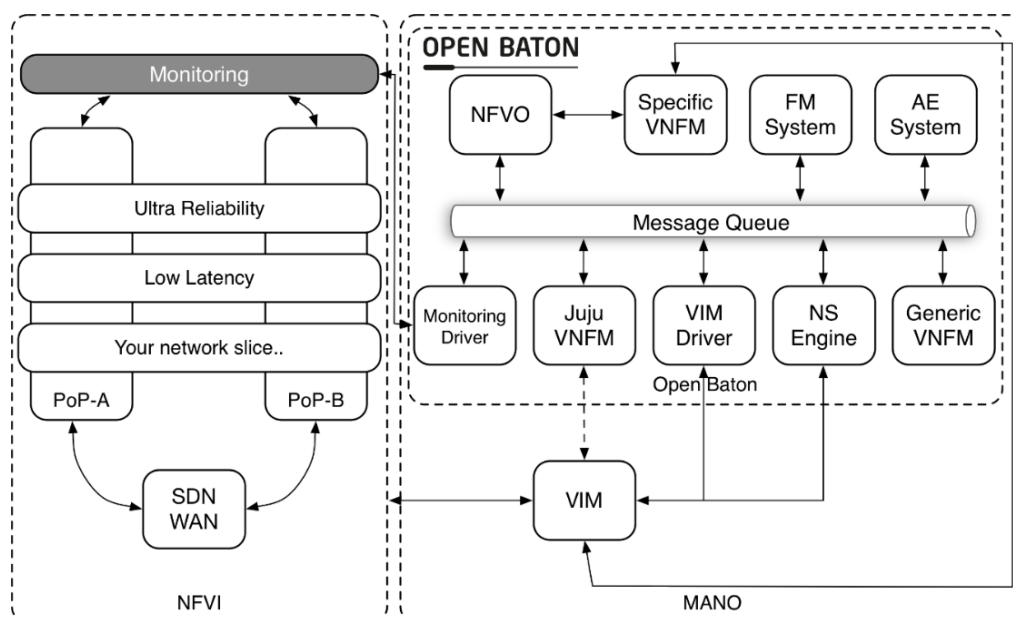


Figure 4 - Open Baton architecture

Current Open Baton release 4 includes many different features and components for building a complete environment fully compliant with the NFV specification. Among the most important are a NFVO following ETSI MANO specification [14], a generic VNFM to deploy Juju charms or Open Baton VNF packages, a marketplace integrated within the Open Baton dashboard, a driver mechanism supporting different types of VIMs without having to re-write anything in the orchestration logic, and a powerful event engine for the dispatching of lifecycle events execution. Finally, Open Baton is included as a supporting project in the project Orchestra6 [16]. This OPNFV initiative seeks to integrate the Open Baton orchestration functionalities with existing OPNFV projects in order to execute testing scenarios (and provide feedback) without requiring any modifications in their projects.

2.1.3. Open Networking Automation Platform (ONAP)

The Linux Foundation's Open Network Automation Platform (ONAP) [17] is the most broadly supported orchestration initiative. ONAP originated in early 2017, based on combination of ECOMP (Enhanced Control, Orchestration, Management & Policy) from AT&T, and Linux Foundation's Open-O (Open Orchestrator) projects.

ONAP provides a comprehensive platform for real-time policy-driven orchestration and automation of physical and virtual network functions (VNF) as well as the capabilities for designing, creating, orchestrating and handling of the full lifecycle management of VNFs, Software Defined Networks (SDN), and the services that all of these things entail. ONAP aims at simplifying the design, creation, orchestration, monitoring, and life cycle management of VNFs, SDNs, and higher-level services. In essence, ONAP is the platform above the infrastructure layer that automates the network. ONAP allows the end users to connect products and services through the infrastructure and allows deployments of VNFs and scaling of the network, in a fully automated manner. ONAP supports scaling of NFV over OpenStack using automated orchestration. The platform coordinates with virtual infrastructure managers (VIMs) such as OpenStack to provide application and tenant connectivity within and between clouds.

The ONAP Platform enables product-independent capabilities for design, creation and lifecycle management of network services. ONAP uniquely provides a unified operating framework for vendor-agnostic, policy-driven service design, implementation, analytics and lifecycle management for large-scale workloads and services. With ONAP, network operators can synchronously orchestrate physical and virtual network functions. This approach allows operators to leverage existing network investments; at the same time, ONAP's openness and ubiquitous acceptance by major network providers around the globe accelerates the development of a vibrant VNF ecosystem. As part of each release, the ONAP community also defines blueprints for key use cases, such as 5G, BBS, CCVPN, Voice over LTE (VoLTE), and vCPE, which the user community expects to pursue immediately. Testing these blueprints with a variety of open source and commercial network elements during the development process provides the ONAP platform developers with real-time feedback on in-progress code and ensures a trusted framework that can be rapidly adopted by other users of the final release.

ONAP consists of several software subsystems. These subsystems are part of two major architectural frameworks:

- a design-time environment to design, define and program the platform
- an execution-time environment to execute the logic programmed in the design phase.

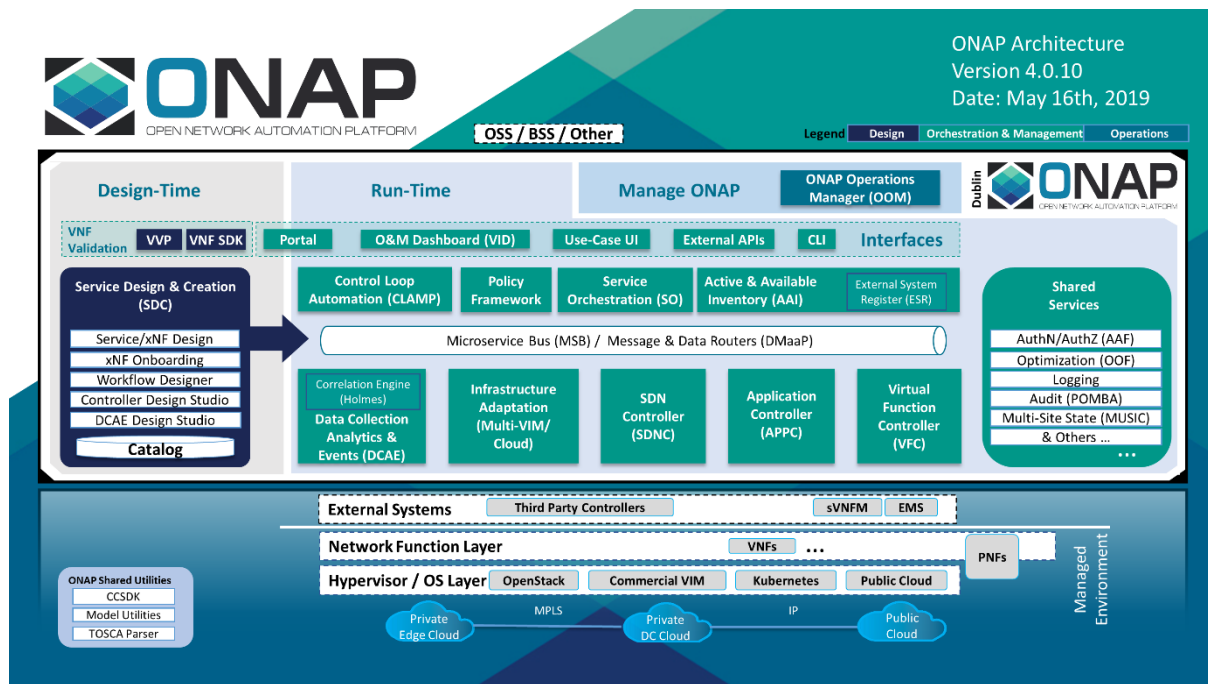


Figure 5 - ONAP Platform Architecture [17]

As a cloud-native application that consists of numerous services, ONAP requires sophisticated initial deployment as well as post-deployment management. The ONAP Operations Manager (OOM) is responsible for orchestrating the end-to-end lifecycle management and monitoring of ONAP components. It is integrated with the Microservices Bus, which provides service registration/discovery and support for internal and external APIs and key SDKs. OOM uses Kubernetes to provide CPU efficiency and platform deployment. In addition, OOM helps enhance ONAP platform maturity by providing scalability and resiliency enhancements to the components it manages. The platform provides tooling for service designers as well as a model-driven run-time environment, with monitoring and analytics to support closed-loop automation and ongoing service optimization. Both design-time and run-time environments are accessed through the Portal Framework, with role-based access for service designers and operations personnel. All these subsystems rely on “common services” to provide access control, logging, data management, and other support. For more on ONAP Policy Architecture the reader is to Annex 3 – ONAP Policy Architecture.

2.1.4. Cloudify

Cloudify [18] is an open-source end-to-end orchestration framework, using the ARIA TOSCA [19] (Topology and Orchestration Specification for Cloud Applications) project as its core. Cloudify consists of a core engine responsible for the lifecycle management of applications and network services, and a set of plugins providing integration points for all needed components—from infrastructure (Compute, Storage, Network) to logging and monitoring so it qualifies as a CMP (Cloud Management Platform) more than an NFVO as this approach was not originally considered for the Cloudify platform.

In terms of where Cloudify fits into the ETSI NFV MANO architecture, it provides a network functions virtualization orchestrator (NFVO) and a virtual network functions manager (VNFM),

although it can also serve only as a generic VNFM and all components can be deployed individually or together. Cloudify is a key orchestration component of the OPEN-O project (ground base for ONAP [17]).

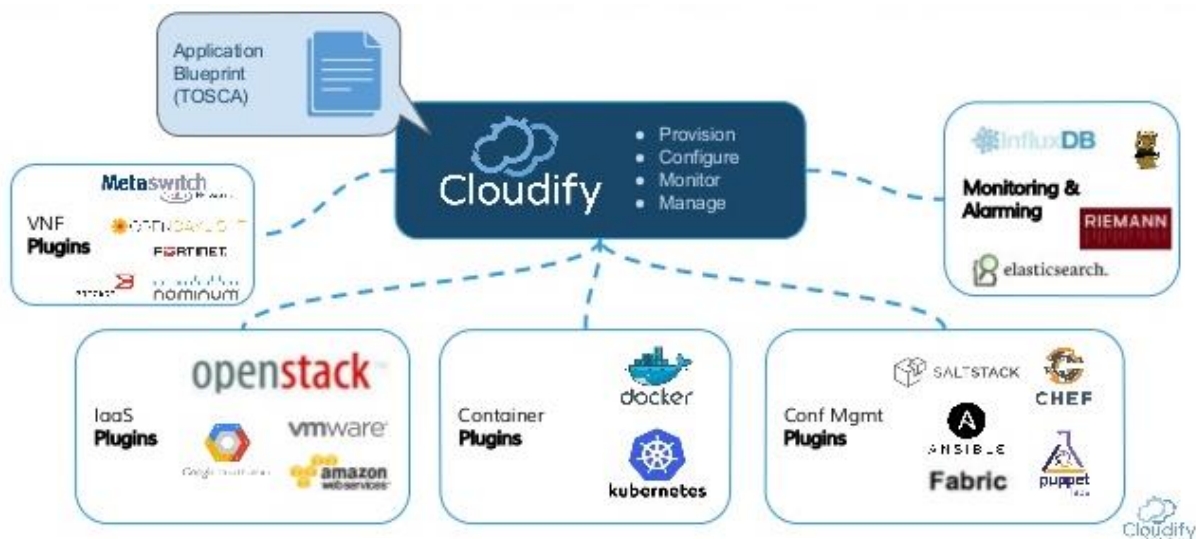


Figure 6 – Cloudify Pluggable Orchestration [20]

Cloudify cloud management platform, according to its specifications [18] includes:

- *VNF & Service Modelling* – VNF modelling enables the user to describe the complete network service with all its resources: infrastructure, functions, service chaining, application code, scripts, configuration management, metrics, and policies, in a generic, descriptive language based on TOSCA and the Cloudify DSL.
- *Orchestration and Workflows* – The Orchestration is the core of the Cloudify Platform. It enables the user to maintain the complete life cycle of the service, from onboarding and instantiation to operations such as scaling, healing, maintenance, updates, and termination.
- *Pluggability* – Pluggability is one of the cores, unique features of Cloudify. It provides reusable components abstraction for the system. The user can model anything in a descriptive language, for example, IaaS, clouds, configuration management tools, SDN components, NFV components, and so on. Cloudify includes a number of officially supported out-of-the-box plugins, but users can also build their own.
- *Security* – Security, in the context of a Cloudify Manager, means securing communication with the Cloudify Manager and controlling who has permissions to use it to execute various operations. Secured communication is achieved using SSL, which enables clients to validate the authenticity of the Cloudify Manager, and to ensure that the data sent to and from it is encrypted.
- *Native Cloud Experience* – An abstraction layer provides access to a full set of features for each cloud environment.
- *Cloud Portability* – Cloudify is technology agnostic and it can be deployed in any cloud environment.
- *Faster App Roll-out* – Lifecycle automation shortens deployment time from days to minutes.
- *Cost Efficient* – Manage costs and optimize cloud usage, from start to finish, with the Cloudify Management Console.

The NFV use case for a Cloudify platform was not originally considered, although, Cloudify offers now a pluggable and extendable architecture and an embedded workflow engine that enables arbitrary NFV service provisioning. However, the decision-making process regarding the overall platform strategy is not flexible at all, as it is controlled internally by Cloudify.

2.1.5. SONATA

Apart from the big players in the NFV environment we have considered other research activities focused on the development of orchestration policies.

SONATA Service Platform is an open-source Orchestrator developed by 5GTANGO [21], a 5GPPP Phase2 Innovation Action project. The SONATA is composed of three main components: Service Platform (SP), Service Development Kit (SDK) and a Validation and Verification (V&V) platform service but it can also be used stand-alone.

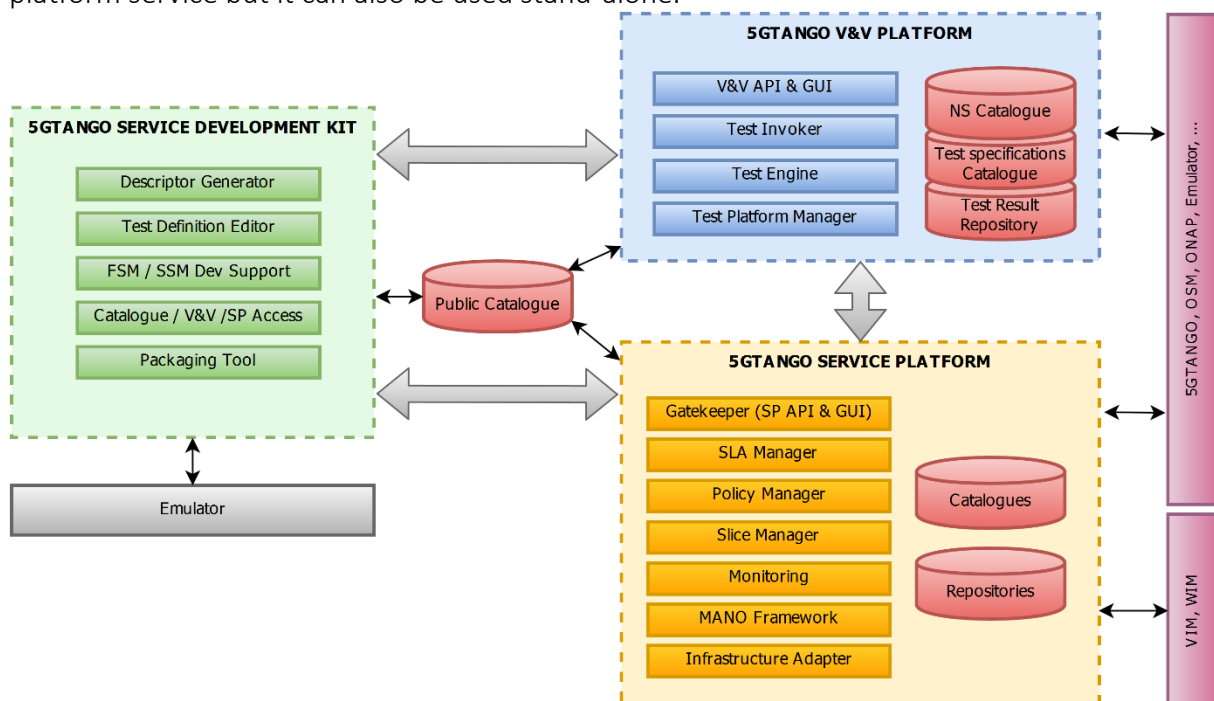


Figure 7 – Main SONATA modules

The platform is following the modular approach for each component, so it combines flexibility and robustness. The main modules are:

- Gatekeeper: controls and enforces whoever (and whatever) wants to interact with the SP and guarantees the quality of the submitted packages
- Catalogues, stores and manages package files
- Repositories to store and manage services
- MANO Framework: the orchestrator, who manages each service's lifecycle, including when the service and/or its functions bring specific managers with them to be used in certain segments of their lifecycle
- Infrastructure Abstraction: hides the complexity and diversity of having to deal with multiple VIMs and WIMs.
- Monitoring provide the telemetry information about the services and function instances

- Policy Manager: define policy rules based on metrics or infrastructure resources utilization
- SLA Manager: define SLAs with certain objectives to be guaranteed by the service provider to the end-
- Slice Manager: define Network Slice Templates by using multiple Network Services (NSs) interconnected, and is able to instantiate, terminate, and perform other advanced Slice management operations such as update, scale, etc.

2.2. Virtual Infrastructure Manager (VIM)

ETSI ISG NFV has defined in its specification the term NFV Infrastructure (NFVI) to denote the infrastructure that provides the enablers to support the instantiation and operation environment for VNFs or CNFs chained together in a forwarding graph. The NFVI is a key component of the NFV architecture that describes the hardware and software components on which virtual networks are deployed.

The Virtual Infrastructure Manager (VIM) is responsible for controlling and managing the compute, storage and some of the networking resources of the NFVI. Specifically, the VIM handles the lifecycle of the VNFs or CNFs, their network requirements (internal and external), provides them with the necessary compute and storage resources as well as manages security to ensure access control. The VIM handles other functions as well, such as collecting performance and fault information, managing software images and virtualized catalogues and more depending on the chosen platform. There can be multiple instances of VIM in the same NFVI and even different between them, in order to have the required capabilities and more.

Nowadays, there are a few options of VIMs in the market, some of them more mature than others and some in more experimental condition. In the following section, more information will be provided for the three VIMs that are to be used from 5Genesis: OpenStack, OpenNebula and Kubernetes.

2.2.1. OpenStack

OpenStack [22] as an open source project that has greatly simplified the path to virtualization. ETSI and OPNFV have defined specifications and released reference platforms for NFV that select OpenStack as the Virtualisation Infrastructure Manager. Additionally, OpenStack is the dominant choice for quite a few management and orchestration functions. NFV on OpenStack offers an agile, scalable, and rapidly maturing platform with compelling technical and business benefits for telecommunications providers and large enterprises. Examples of such benefits:

- Standardised interfaces between NFV elements and infrastructures are provided
- Resource pools available cover all network segments
- Network and element deployment automation, providing roll-out efficiency
- Pluggable architecture with documented APIs, UIs, shared services, operations, automation
- All popular open source and commercial network plug-ins and drivers are available
- Outstanding global community contributing to rapid pace of innovation, working on unique NFV requirements from users, related communities and standards developing organizations, NFV features in every release since 2013.

- Proven Telecom as well and enterprise implementations: AT&T, China Mobile, SK Telecom, Ericsson, Deutsche Telekom, Comcast, Bloomberg, and more.

OpenStack is inherently a multi-tenant platform where multiple users on the same cluster share compute, storage and networking resources without awareness of other users. Tenant networks are isolated from each other, which is achieved through the Neutron service that provides each tenant their own network namespace by leveraging either VLAN segregation or VXLAN/GRE tunnelling based overlay networks.

2.2.2. OpenNebula

OpenNebula [23] is an open-source management platform to build IaaS private, public and hybrid clouds and manage heterogeneous distributed data centre infrastructures. It was started as a research project in 2005, setting as goals to create efficient solutions for managing virtual machines on distributed infrastructure, with the ability to scale at high levels. The first public release of the software occurred in 2008, while 2019 was the year that the latest release, release-5.8, was published.

OpenNebula provides features at two main layers of Data Centre Virtualization and Cloud Infrastructure:

- Data Centre Virtualization Management: OpenNebula integrates directly with hypervisors (like KVM) and containers (like LXD), and has complete control over virtual and physical resources, providing advanced features like Capacity Management, Resource Optimization, High Availability and Business Continuity.
- Cloud Management (VIM): OpenNebula also provides a multi-tenant, cloud-like provisioning layer on top of virtual infrastructures, including existing infrastructure management solutions (like VMware vCenter). It provides provisioning, elasticity and multi-tenancy features including virtual data centres provisioning, data centre federation and hybrid cloud computing to connect in-house infrastructures with public clouds.

2.2.3. Kubernetes

Kubernetes [24] is an open-source system for automating deployment, scaling, and management of containerized workloads and services that facilitates both declarative configuration and automation. It is possible to launch several containers grouped together in an entity called a *pod*. A pod generally represents one or more containers that should be controlled as a single application. A replication controller ensures that a specified number of *pod replicas* are running at any one time and are always available, providing stability. Another feature provided is a Kubernetes Service. This service is an abstraction which defines a logical set of pods and a policy by which to access them - sometimes called a micro-service.

Kubernetes as a VIM is a fairly new concept due to a number of networking issues and isolation concerns that need to be addressed before its utilization. However, using Kubernetes brings the advantage of being able to deploy and use certain VNF types in Cloud-native formats. Cloud-native technologies empower organisations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds.

3. MANO SOLUTION IN 5GENESIS

3.1. System specifications

As per the specifications of the 5GENESIS Management and Orchestration layer in Deliverable D2.2 [5], the MANO layer consists of three major components: (1) **Slice Manager**, (2) **Network Function Virtualisation MANO** and (3) **Network Management System**, as depicted in Figure 8, although in this deliverable we will not cover the Slice Manager [27].

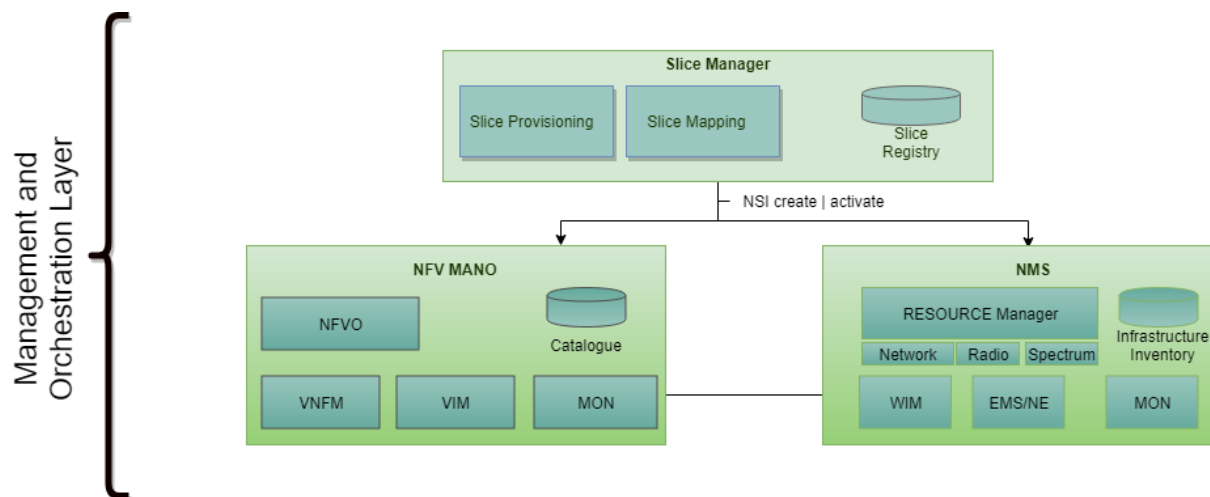


Figure 8 - MANO layer components

The Network Function Virtualisation Management and Orchestration (NFV MANO) is responsible for the virtualised components, so it manages the Network Services (NS) and therefore Virtualised Network Functions (VNF). The NFV MANO Catalogue component is connected to the Coordination layer through a second northbound interface to onboard the NS descriptors.

The underlying non-virtual resources, such as transport network and radio elements, are controlled by the Network Management System (NMS).

3.1.1. NFV MANO (Network Function Virtualisation Management and Orchestration)

The NFV MANO components submits requests to the VIM to instantiate Network Services that consist of VNFs within specific infrastructures for computing, storage and network virtualisation. Through a northbound interface, NFV MANO can receive requests for already onboarded Network Services that are stored in the NS catalogue.

The **Virtualisation Infrastructure Manager (VIM)** manages the virtualised infrastructure. This covers management tasks regarding the NFVI physical and virtual resources (and the mapping between the two), VNF forwarding graphs, VNF lifecycle performance, as well as the discovery of features to optimize resources. From the MANO components, the VIM will interact directly to the Infrastructure layer, interfacing with the virtual infrastructure [28].

The **NFV Orchestrator (NFVO)** orchestrates the NFVI resources across multiple VIMs. This task may, when necessary, involve the WAN Infrastructure Manager (WIM, in the NMS). In addition,

the NFVO manages the lifecycle of Network Services. For Network Services that are distributed across two or more NFV infrastructures, the NFVO can interface directly with the WIM to orchestrate the provisioning of resources.

The **VNF Manager (VNFM)** is the lifecycle manager for VNFs that are controlled by the NFVO. The VNFM instructs the VIM to instantiate, scale, update, upgrade and terminate VNFs. In the context of the 5G architecture in 5GENESIS, it is important to establish the interface to the EMS (in the NMS) for each virtual and physical network function.

3.1.2. NMS (Network Management System)

Through the Network Management System (NMS), network administrators manage network resources within the infrastructure layer, where independent network components (software and hardware) are managed inside a bigger network management framework. A special use case for the NMS framework is when the network configuration is performed manually or through a proprietary interface, for instance for once-off resource configuration for static provisioning or in cases where the resource-specific programmatic interfaces are not available in the Network Controller.

In the NMS, the **Element Management System (EMS)** component provides the fault, configuration, accounting, performance and security (FCAPS) management of physical and virtual network functions, where different EMS instances will likely be used to manage different radio access technologies with varying management and monitoring capabilities. The EMS may include the management of 5G core network functions and radio elements.

The **Resource Manager** component keeps track of the available platform resources and updates the Infrastructure Inventory. It communicates with the individual Element Managers to configure resources that are requested for slice creation.

The **Wide-area Infrastructure Manager (WIM)** manages the network between the NFVIs, the infrastructure gateway and the radio edge. Because of the distributed nature of this network and the network services (across the SDN-based WAN, across different core network domains and across different Points of Presence) an updated VIM registry is required in each domain. Communication through a direct interface to the NFVO or indirectly through the NMS is supported.

3.1.3. MANO Requirements

This section summarises the basic requirements that the 5GENESIS MANO Platform need to cover, as described below in

Table 2.

The Functional requirements identified in the MANO layer, has been listed in D2.1 [29], based on this information we defined the specification and functionalities to be covered by task 3.1 in the different platforms. Some of the basic MANO functionalities like managing the lifecycle of the network services have been delegated to the Slice Manager [27] to allow more flexibility and control when running the slice.

Table 2 - MANO requirements

User Requirement	Technical specification
Resource Catalogue	Interfaces to list the NSD (Network service Descriptor Specification) (i.e. ETSI SOL 004)
Adaptation of Services	Scale up/down services for optimization
Flexible and Fast Allocation of Network Resources	KPI defined for service creation time in less than 90 min
Distributed NFVI on User or Service Demand	Service placement based on ITU requirements
Network Service Composition	Composition of e3e services with lifecycle control
NFV Management and Organisation	Lifecycle management of services following specification (i.e. ETSI NFV ISG)
MEC Management and Organisation	Lifecycle management of applications in the MEC by NFVO

3.2. MANO architecture

In this section, the key components of the NFV MANO architecture are introduced, according to standard defined by the European Telecommunications Standards Institute (ETSI) in 2014 [30]. The blue delimited block of Figure 9 shows the NFVO, VNFM, and VIM entities, explained below. Other functional blocks are also introduced, such as the NFV Infrastructure (NFVI) that comprises hardware and software resources, and the Monitoring entity.

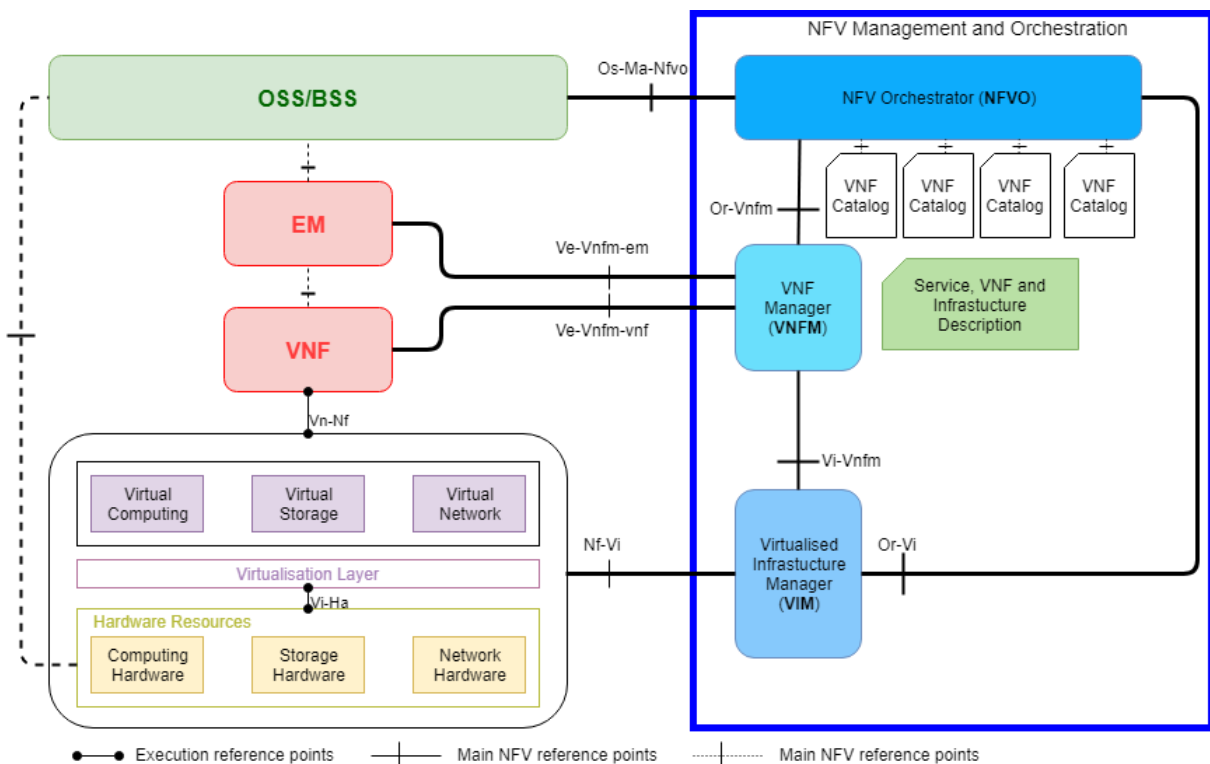


Figure 9 - NFV MANO architecture

3.2.1. NFV Orchestrator (NFVO)

The NFV Orchestrator [30] is a key component located at the highest hierarchical level of the NFV MANO architecture [31]. The NFVO is in charge of the orchestration and management of NFVI resources, as well as the requests to create through the VIM and lifecycle management of Network Services (NSs), and the respective validation, authorisation, and management on the NFVI. Some NFVO operations are:

- Onboarding of Network Services (NSs) and VNF packages
- NSs instantiation
- Management of the VNFs instantiation
- Policy management for NS and VNF instances
- Collect information on NFVI resources

The NFVO works jointly with the VNF Manager to ensure that the service provided fulfils the quality requirements. The NFVO is able to interact with the resources managers of the infrastructure needed to deploy a service. In the case of Open Source MANO, the component in charge of this interaction is the Resource Orchestrator (RO). “The Resource Orchestration Engine is responsible for managing and coordinating resource allocations across multiple geo-distributed VIMs and multiple SDN controllers. The VIM and SDN Plugins are responsible for connecting the Resource Orchestration Engine with the specific interface provided by the VIMs and SDN controllers.”[32]

3.2.2. VNF Manager (VNFM)

The VNF Manager [30][31] is responsible for the configuration and supervision of the VNFs’ lifecycle, and it performs the coordination and adaptation needed for configuring and event reporting between the VIM and the EM. Some VNFM operations are:

- VNFs instantiation
- VNFs up/down and in/out-scaling
- VNFs updating and/or upgrading
- VNFs termination

Indeed, CRUD operations are supported. Multiple VNFMs may be deployed, and one VNFM may serve a single VNF or multiple VNFs. All VNF instances ideally have an associated VNFM.

3.2.1. Virtualisation Infrastructure Manager (VIM)

The VIM [30][31] is in charge of the control and management of interactions of the VNF resources (computing, storage, and network). It creates and assigns the virtual resources needed by specific functions, performing the virtualization of those resources. Usually, the resources are located within the NFV Infrastructure (NFVI).

The VIM performs:

- Resource management
 - o Orchestration of NFVI resources
 - o Management of the virtualised resources capacity
- Operations
 - o Analysis of performance issues from NFV Infrastructure

- Collection of information related to fault and planning, monitoring, and optimization capacity

Multiple VIM instances may be deployed. A VIM may be specialized in handling a specific NFVI resource (for instance, networking (WIM)), or multiple types of resources.

3.2.2. Monitoring

Efficient and comprehensive monitoring of services is a fundamental aspect in an NFV environment [30][33]. The monitoring solution should be able to cope with VNF migration, auto-scaling and multi-tenancy of VNFs.

There are two levels of abstraction to differentiate. VIM-level monitoring collects metrics and events from the NFV Infrastructure, both physical and virtual resources. For instance, OpenStack's Ceilometer module is in charge of collecting these measurements. The orchestrator is able to poll directly these metrics through the correspondent API, this is Orchestrator-level monitoring. There are several aspects that must be considered, such as delay, polling frequencies or the importance of metrics in each case.

The times of detection and notification are key, especially in critical node services where the latency should be under 50 ms., although other services admit higher latencies such as seconds or 10s of seconds.

Another aspect to be considered is the trade-off between time-frequency and stored data, so called polling frequency. It depends on the monitored parameter and its relevance to the system, as well as the provided service. For instance, latency is critical in URLLC use cases, but not in mMTC use cases.

The events describe the resources change of state in a monitored function; for example, starting a virtual machine instance. In addition, alarms provide the rules needed to notice when a state transition occurs.

3.2.3. Catalogues

The repositories holding the information in an NFV system are an important part in the architecture. There are two catalogues and two repositories.

A VNF catalogue is a repository of all the available VNF Packages. These packages are composed of the VNFD and the manifest file, among others; and it should be accompanied by the software image. It is provisioned, deployed and managed by the NFVO and the VNFM.

A VNF Descriptor (VNFD) [34][2] is a template which defines the deployment and behaviour requirements of a VNF, containing some network resources requirements. The VNFDs are stored in the VNF catalogue, further explained in Section 3.2.3 and an example in Annex 1 – VNF Descriptors. The VNFM has access to a repository with the VNF packages, mainly represented via their VNFD. Thus, it can create instances of the VNFs described and manage their lifecycle.

Each VNFD, based on the TOSCA standards [35], must include the Virtual Deployment Units (VDUs), Connection Points (CPs) and Virtual Links (VLs). There are also optional components such as floating IP or descriptions.

- The VDU is a basic part of the VNF where the VM that hosts the network function is described.

- The CPs connect the virtual links to a VDU. A CP must have a virtual link and a virtual binding associated.
- The Virtual Links provide connectivity between different VDUs through the CPs.
- The floating IPs are IPs used in order to provide access to VDU from public (external) network.
- It is possible to define multiple node types in a VNFD. For instance : VDU1, VDU2, CP1, CP2, VL1, VL2, etc.

Figure 10 shows an example of a VNFD template graph in OSM in which VDU, CP and internal VL are represented together with their connections.

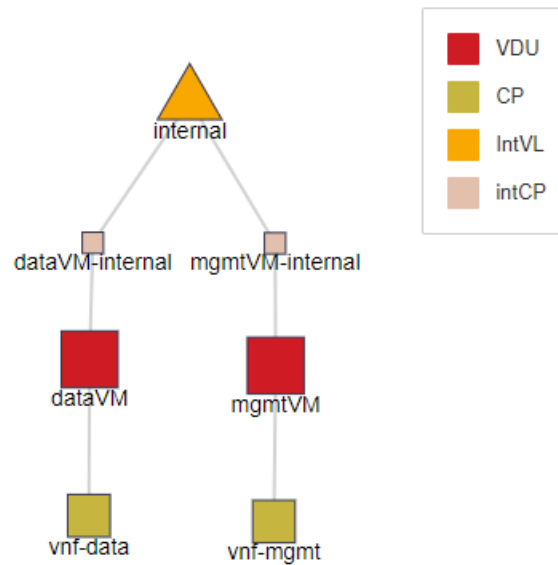


Figure 10 - VNFD template forwarding graph

A Network Services (NS) catalogue is a repository of the usable. These NSs support the creation and management of the deployment templates that mainly consist of a Network Service Descriptor (NSD) and a Virtual Link Descriptor (VLD) [36][2]. An NSD describes network information used by the NFVO to instantiate a network service. It contains a list of the VNFs that compose the Network Service and a list of Virtual Links that are referenced by the VNFD in order to define the network connectivity. There are optional parameters, such as the list of dependencies between different VNFs. Examples of NSD can be found in Annex 2 – NS Descriptors and a graphical representation of a NS forwarding graph in Figure 11, with the constituent VNFs, VLs and CPs.

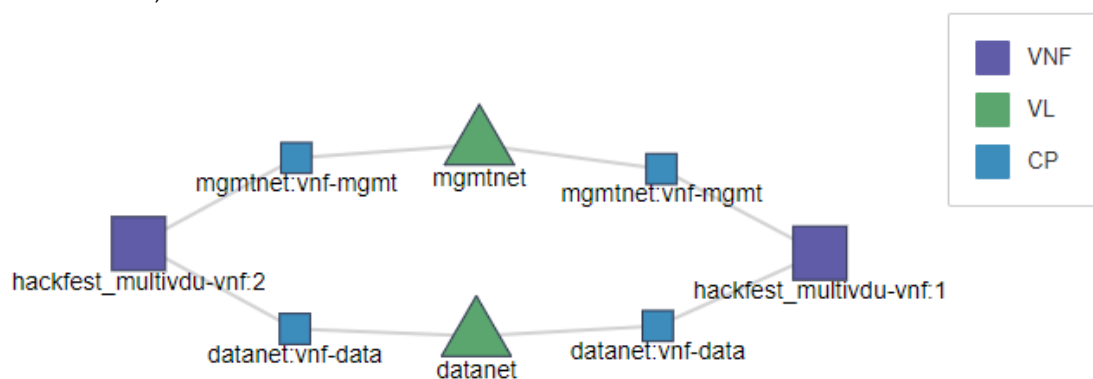


Figure 11 - NS forwarding graph

The NFV Instances repository holds information of all VNF and NS instances that are represented by a VNF/NS record, which is updated during the lifecycle. The NFVI Resources repository holds information of the resources available, reserved, and allocated.

3.3. Role in 5GENESIS architecture

The NFV MANO and the NMS components are located in the MANO Layer according to the 5GENESIS architecture (Figure 12). The NFV MANO is in charge of the management and orchestration of all resources in the cloud data centre: this includes computing, networking and storage.

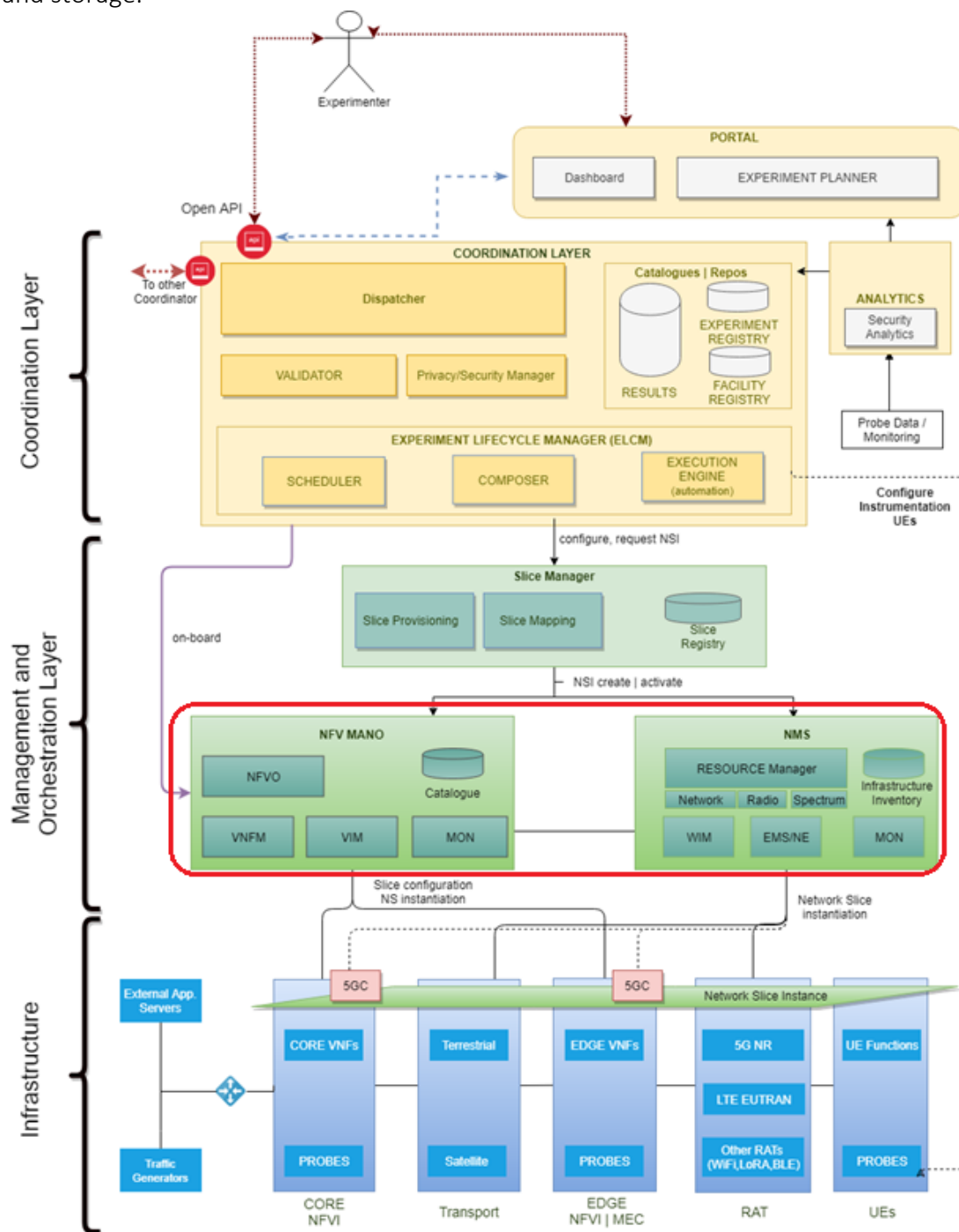


Figure 12 – NFV MANO and NMS in the 5GENESIS reference architecture

The NFV MANO allows flexible on-boarding, sitting between the Slice Manager and the physical infrastructure, solving the problems associated with the rapid span up of network components. In 5GENESIS, this module holds the VNF and NS catalogues, the NFVO, the VNFM and the VIM. Those components allow the management of the main datacentres and optionally, edge infrastructure. Besides a monitoring tool to gather information and statistics about the lifecycle of the network services.

Some platforms include an edge infrastructure aside from the core NFVI, which according to ETSI GS MEC 003 [37], can be managed by a centralised NFVO without the need of having a secondary edge orchestrator (MEO) managing the edge infrastructure.

5GENESIS NMS is, on command by the Slice Manager, addressing the network configuration external to the services deployed by the NFVO. It is a common component to all the platforms but each one of them has different requirements and configurations. This is due to the fact that each platform implements completely different infrastructures.

The requirements that need to be fulfilled by the 5GENESIS NFV MANO and the NMS are described in the previous section.

Figure 13 below describes a flow diagram representing how the MANO components interact with the rest of the platform.

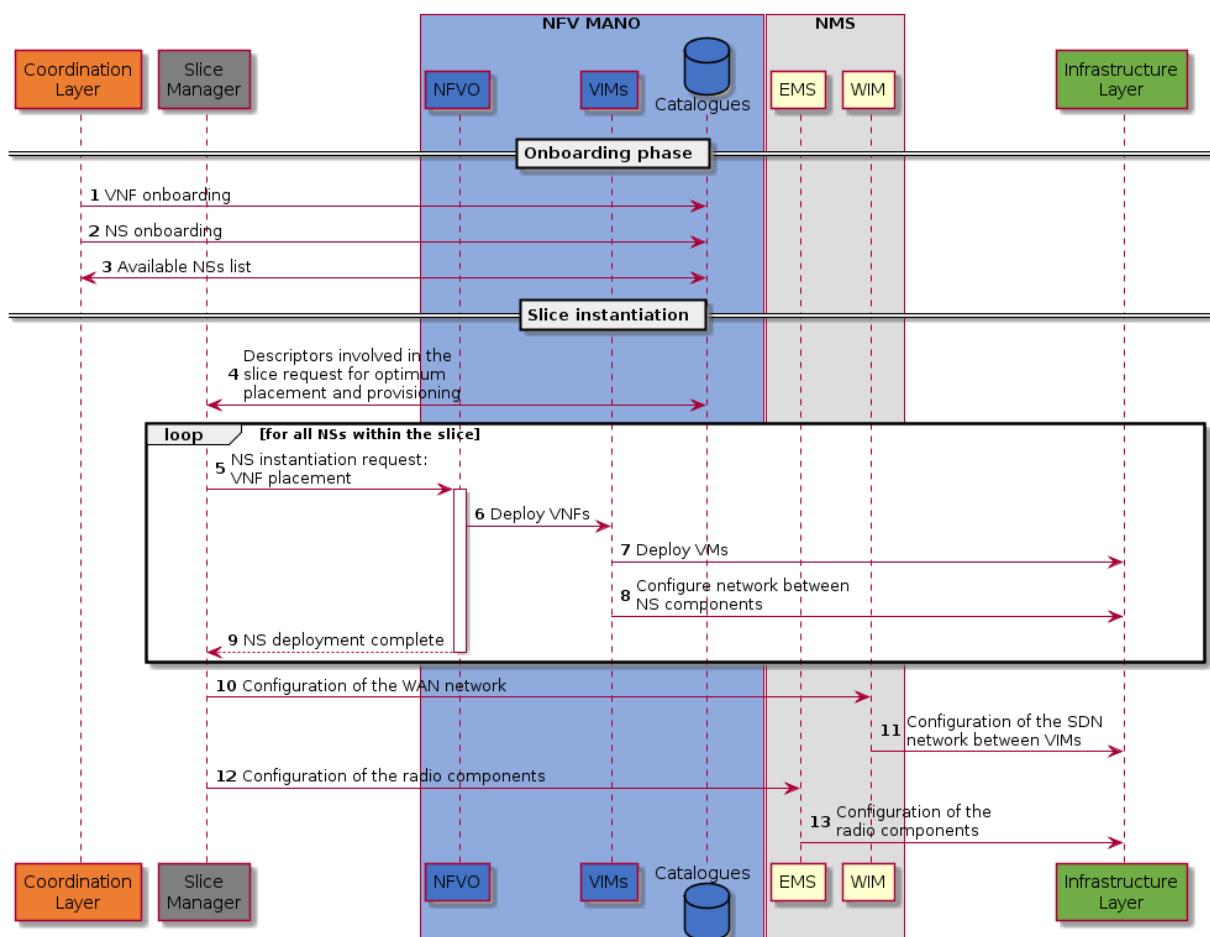


Figure 13 - NFV MANO Flow and interactions diagram

3.3.1. Interfaces

The NFV MANO and the NMS components, as seen in below in Figure 14, have several interfaces with their neighbour components: The Coordination Layer (*Co-Ma*), the Slice Manager (*Sm-Ma*), which also communicates with the NMS through *Sm-Nm*, and the infrastructure (*Ma-In*).

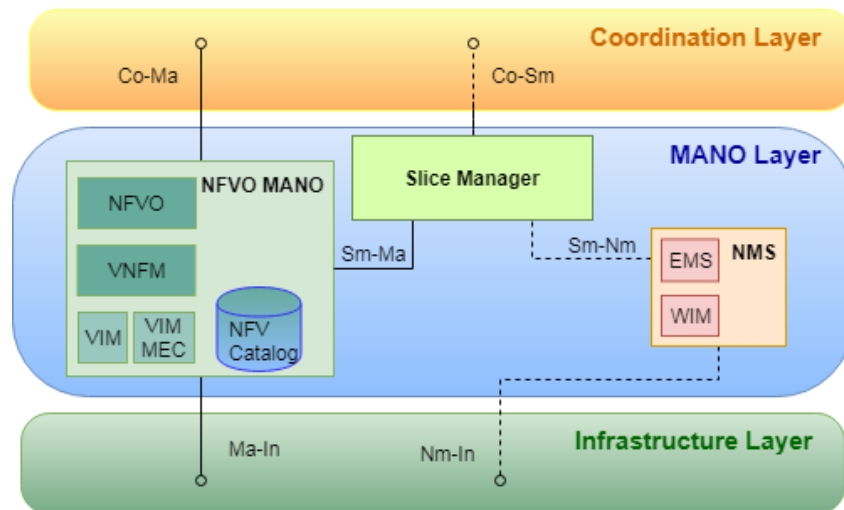


Figure 14 - NFV MANO and NMS interactions

High level description of the interfaces

- Coordination Layer - NFV MANO (*Co-Ma*)**
 The interface is used to perform CRUD operations over the VNF and NS catalogues from the Coordination Layer. That is: onboard VNF referred in the NS descriptors that will later be used for the experiments, delete, update and consult them.
 Another feature of this interface is to collect monitoring information about the processes happening inside the NFV MANO for statistics and analytics
- Slice Manager - NFV MANO (*Sm-Ma*)**
 Through this interface the Slice Manager controls the NFV MANO as a tool to realise the slicing. The Slice Manager creates tenants and registers different VIMs to carry out the isolation between slices and uses this interface too to manage the lifecycle of the different Network Services within the slice, using the tenants and VIMs previously created.
- Slice Manager - Network Management System - (*Sm-Nms*)**
 This interface is used for the Slice Manager to configure the WAN and the radio network to connect the external network resources and UEs to the Network Services and to manage the connection between the different VIMs.
- NFV MANO - Infrastructure Layer (*Man-Inf*)**
 The virtual infrastructure is managed from the MANO VIMs, which decides where and how each service is deployed and connected.
- NMS - Infrastructure Layer (*Mms-Inf*)**
 The NMS acts as an intermediary, adjusting Slice Manager commands towards the infrastructure to configure network parameters as described in the workflow in Figure 13.

3.3.2. Roadmap

Even when all platforms are different, and the technologies selected could vary, the 5GENESIS MANO component needs to have common features and behaviour. To homogenize such behaviour, a list of generic action points (A, in the table), milestones (M) and subtasks (s) has been defined in Table 3.

Table 3 - MANO roadmap

#	Task
A.1	Deployment and configuration of MANO
A.2	Deployment and configuration of core VIM
S.2.1	Integration Main DC VIM - NFVO
S.2.2	Test NS deployment in cloud infrastructure (with manual placement)
A.3	Deployment of Edge DC VIM
S.3.1	Communication NFVO - core VIM - edge VIM
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)
M1	NS with manual placement deployed in both PoPs (Main and Edge)
A.4	WIM development and configuration for connecting VIM PoPs
S.4.1	SDN network configuration
A.5	NSD/VNFD operations
S.5.1	VNFD/NSD onboarding
S.5.2	VNFD/NSD validation
S.5.3	Integration with the Portal
A.6	Integration with Slice Manager
S.6.1	Slice provision (Core/Edge by Experiment template)
S.6.2	Deployment of a NS through the Slice Manager
M2	Slice provision and deployment (preliminary) in the NFVI through SM
S.6.3	Automatic placement of a full NS through the Slice Manager
A7	Monitoring of services /virtual infrastructure
S.7.1	Monitor individual VNF
S.7.2	Monitor slice (aggregated monitoring)
S.7.3	Alert received based on Analytics module (coming from T3.3)
S.7.4	Endpoint NFVO configuration
S.7.5	RuleBook- Policy engine (automation)
M3	Slice e2e multidomain deployment with automatic scaling based on analytics

Each platform has its own distribution of tasks per phase, among the three cycles agreed for the 5GENESIS Platform deployment:

- Phase 1: Sept. 2018 – June 2019
- Phase 2: June 2019 – March 2020
- Phase 3: March 2020 – Dec. 2020

Some platforms may introduce extra action points, as each one might have different requirements depending on the physical infrastructure and the use cases that are going to be demonstrated.

In section 4, we will review the status of these tasks on a per platform basis, up to the submission of this deliverable.

4. RELEASE A SUMMARY AND FUTURE PLANS

This section describes how the MANO solution designed for 5GENESIS is being particularised for each platform, dividing the tasks of the roadmap explained in section 3.3.2 between the three phases, following to their own schedule and needs.

As part of the information provided for each platform, we have used a colour code for indicating the current status of the activities:

- Green: task completed
- Yellow: task in progress
- Red: task not started yet

According to this code, a certain task could appear in colour red even when the task might be on schedule or be programmed for different phases in different platforms.

4.1. Malaga platform

4.1.1. MANO solution adopted and features

Malaga platform will be testing three 5G use cases: 3GPP MCS (Mission Critical Services) deployed at the Core NFVI, MCS deployed at the MEC, and enhanced surveillance video. The three use cases consider the services to be deployed as virtualised Network Services. The introduction of the MEC mechanism for these use cases is critical. Indeed, it reduces end-to-end latency drastically and most of the times this improvement is almost directly mapped to an enhancement of overall service KPIs and end-users QoE.

Under certain crowded or loaded conditions, only the fact of involving a MEC-server may not be enough, and a split of the traffic could be the most appropriate option to achieve the best overall performance.

To build this scenario, among all the MANO alternatives studied in section 2, the Malaga Platform has opted for the combination **OSM-Openstack-OpenNebula**: OSM [5] as the NFVO and generic VNFM, Openstack [22] as the Main DC VIM and OpenNebula [23] as the edge DC VIM.

OpenNebula has been selected as edge VIM for the following reasons:

- Stores disk images in the local storage as “golden images” prepared and parametrised for each service, allowing a shorter boot time.
- Uses VLAN tags to mark NFV traffic, simplifying the network interconnection between VMs.
- Includes an internal monitoring and alert system based on metrics gathered by OpenNebula, being able to trigger scalability rules or alerts.
- Lightweight and easy to install
- Easy to extend through REST APIs

This selection covers the must-have requirements and specifications defined in section 3.1 for each of the components. The expertise of the partners involved in the Platform in these technologies was key to take the final decision.

As already explained in D4.4 [38], Openstack Queens release is used for the Core NFVI management (distributed in three servers), OpenNebula 5.8.1 is used for the Edge infrastructure management (distributed in two servers) and OSM R5 as NFVO. Figure 15 depicts a high-level distribution of the deployment with the different PoPs available in the infrastructure.

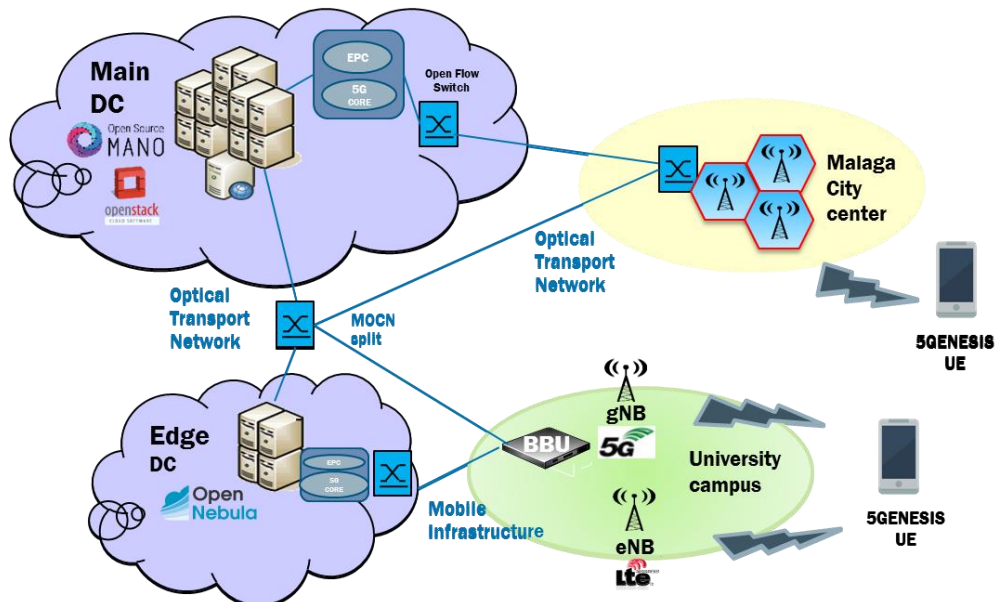


Figure 15 - Malaga Platform MANO deployment











4.1.2. Platform status (Release A)

Following the tasks list proposed in section 3.3.2, Table 4 represents the achievements accomplished by the Malaga Platform, classified by phases.

As it can be seen, the tasks assigned for Phase 1 have already been fulfilled and most of the tasks scheduled for Phase 2, up to the submission of this deliverable.

Table 4 - Achievements Malaga Platform

#	Task	Platform specific	status
A.1	Deployment of NFVO	OSM R5	P1
A.2	Deployment of core VIM	Openstack Queens	P1
S.2.1	Integration Core VIM – NFVO		P1
S.2.2	Test NS deployment in cloud infrastructure (with manual placement)		P1
A.3	Deployment of edge VIM	OpenNebula v5.8.1	P1
S.3.1	Communication NFVO - core VIM - edge VIM		P1
S.3.1.1	Edge SDN controller configuration	ONOS	P1





#	Task	Platform specific	status	
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)	MCPTT	P2	
M1	NS with manual placement deployed in both PoPs (Core and Edge)	MCPTT	P2	
A.5	NSD/VNFD operations		P2	
S.5.1	VNFD/NSD onboarding		P2	
A.6	Integration with Slice Manager		P2	
S.6.1	Slice provision (Core/Edge by Experiment template)		P2	
S.6.2	Deployment of a NS through the Slice Manager		P2	
M2	Slice provision and deployment (preliminary) in the NFVI through SM		P2	
A7	Monitoring of services /virtual infrastructure		P2	
S.7.1	Monitor individual VNF	Prometheus + Node exporter	P2	

4.1.3. Future plans

The subtasks that have not been accomplished yet, and the phase each task is scheduled to be completed, are presented in Table 5. It is yet to be decided whether the Malaga Platform will need a WIM or not but, in any case, it would be a good added feature that big experiments could be deployed using the benefits of both infrastructures (main and edge), having this potential feature as optional in this platform. The development of the WIM could also be extended to include other platform's VIMs, as part of the platform interconnection procedure. As soon as the next versions on the Slice Manager become ready, Malaga Platform will dedicate effort to integrate them and re-test all new and past features in a continuous integration approach.

Regarding the RuleBook (S7.5 in Table 5), in deliverable D2.2 [5] it was already included the idea of exploring Model Based Testing techniques as part of the validation of the experiments. The information from this enhanced validation could be used to build the RuleBook according to the specific SLAs for the slices. Although, its implementation is planned for phase 3, some proposals in that direction has been elaborated in [39], where the alert received based on Analytics plays a key role.

Table 5 - Action plan Malaga Platform

#	Task	Platform specific	status	
A.4	WIM development and configuration for connecting VIM PoPs		P2	
S.4.1	SDN network configuration		P2	
A.5	NSD/VNFD operations		P2	
S.5.2	VNFD/NSD validation		P2	

#	Task	Platform specific	status	
S.5.3	Integration with the Portal		P2	
A.6	Integration with Slice Manager		P2	
S.6.3	Automatic placement of a full NS through the Slice Manager		P2	
A7	Monitoring of services /virtual infrastructure		P2	
S.7.2	Monitor slice (aggregated monitoring)		P2	
S.7.3	Alert received based on Analytics module (coming from T3.3)		P2	
S.7.4	Endpoint NFVO configuration		P2	
S.7.5	RuleBook- Policy engine (automation of actions based on analytics)		P3	

4.2. Berlin platform

4.2.1. MANO solution adopted and features

The Berlin platform adopted Open Baton as the NFV management and network orchestration (MANO) solution. Mainly, the Berlin platform used Open Baton for the deployment and orchestration of NFVs, including virtual instruments interacting with the test automation platform TAP 59 for measurements and system evaluations, and for orchestrating and deploying components of the Open5GCore [<https://www.open5gcore.org>]. A detailed description of Open Baton as the management and orchestration solution for the Berlin platform is presented in D4.13 [44]. Apart from customizing the dependency information required by Open Baton in order to deploy the 5GCore components, extensions developed within 5GENESIS focused in the first project year around providing interaction between Open Baton and TAP.

4.2.2. Extensions developed as part of the Project

5GENESIS chose to employ TAP [40] as a commercial tool to build its Experiment Lifecycle Manager on. For that, the interaction between TAP and Open Baton was necessary to provide functionalities to deploy all required components for the execution of an experiment from TAP via Open Baton. For that, an **Open Baton Keysight TAP8 Plugin** has been developed, automating the provisioning of the VNFs needed for the experiments.

The plugin can create and destroy VM instances, upload and delete VNF packages, create and remove Network Services Descriptors and launch and delete Network Services on Open Baton. The plugin uses HTTP to communicate with Open Baton's REST API. The plugin comprises the following components (c.f. Figure 16):

- Open Baton Instrument
- Open Baton Allocate Step
- Open Baton Destroy Step

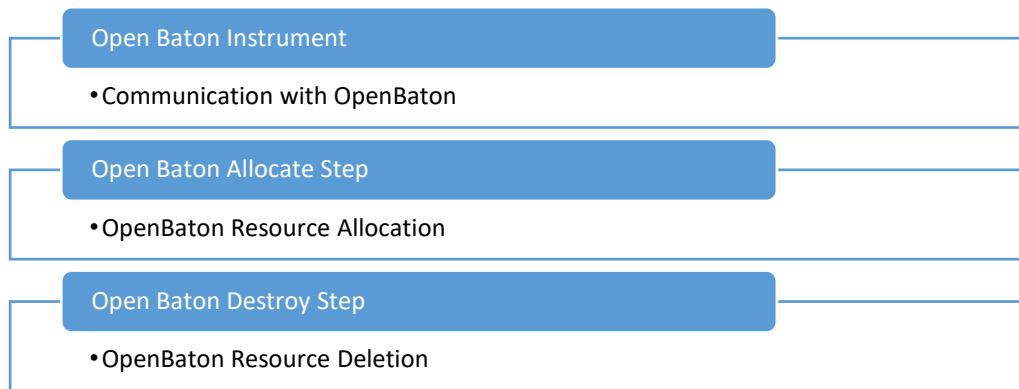


Figure 16 - Open Baton TAP Plugin major components

The **Open Baton Instrument** is a TAP Instrument, used to facilitate the communication with Open Baton's REST API and provides configuration options required for establishing this communication.

For the initial implementation, the creation and destruction of the different Resource types was combined into **two TAP TestSteps: Open Baton Allocate and Open Baton Destroy Steps** respectively. Both TestSteps utilize the Open Baton Instrument to communicate with the Open Baton API.

The Open Baton Allocate step executes the following procedures:

1. Creating a VM for the underlying OpenStack (as deployed in Berlin).
2. Uploading a VNF package provided by the user.
3. Creating an NSD utilizing the VNF create in step 2.
4. Launching an NSR from the NSD.
5. Polling the newly created NSR's status to confirm successful launch.

To optimize the deployment times – a desirable key feature of 5G systems -- the test step supports parallel execution for the deployment of multiple VNFs on the same VM.

The Open Baton Destroy step executes the following procedures:

1. Stop and delete any NSRs created.
2. Delete any NSDs previously created.
3. Delete the uploaded VNFs.
4. Remove the VM instance.

The resources created can be used by other test steps to trigger experiment procedures and gather result data.

4.2.3. Platform status (Release A)

With the integration of Open Baton and the TAP-based components of the 5GENESIS Experiment Lifecycle Manager, the Berlin platform is currently fully capable of deploying and executing full end-to-end experiments.

As illustrated in Figure 17, the developed extensions allowed to integrate and orchestrate the FOKUS site and the IHP site of the Berlin platform and specially to deploy NFVs in various availability zones across the platform.



With that, the Berlin platform allows industry and academia to deploy and execute full end-to-end (5G) experiments given that the experiments are described via TAP test cases. To facilitate the specification of test cases, TAP provides a graphical user interface suitable for users knowledgeable in using professional testing tools.

Table 6 - Achievements Berlin Platform

#	Task	Platform specific	Status	
A.1	Deployment of NFVO	Open Baton	P1	
S.1.1	Evaluation of the API ETSI SOL005		P1	
S.1.2	Implement wrapper for Open Baton north-bound API		P1	
A.2	Deployment of core VIM	Openstack Stein	P1	
S.2.1	Integration Core VIM - NFVO		P1	
S.2.2	Configuring zones for manual placement		P1	
S.2.3	Test NS deployment in cloud infrastructure (with manual placement)		P1	
A9	TAP support in MANO		P1	
S9.1	Implement TAP plugin for Open Baton		P1	

4.2.4. Future plans

To additionally support user preferring to specify experiments and trials via a WEB portal, and to support parallel execution of trials of different users requiring dedicated resources, the orchestrator requires further additions. These encompass:

- Integration with the Resource Manager of the 5G architecture and
- Integration with the 5GENESIS Slice Manager

Which are scheduled for Phases 2 and 3 of the 5GENESIS project respectively. Additionally, FOKUS currently evaluates the option to support OSM as an additional orchestration tool, which would result in a homogeneous orchestration approach across all 5GENESIS platforms.

Table 7 - Action plan Berlin Platform

#	Task	Platform specific	Status	
A.3	Deployment of edge VIM	Openstack	P2	
S.3.1	Communication NFVO - core VIM - edge VIM		P2	
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)		P2	
M1	NS with manual placement deployed in both PoPs (Core and Edge)		P2	

#	Task	Platform specific	Status	
A.4	WIM development and configuration for connecting VIM PoPs		P2	
S.4.1	SDN network configuration		P2	
A.5	NSD/VNFD operations		P2	
S.5.1	VNFD/NSD onboarding		P2	
S.5.2	VNFD/NSD validation		P2	
A.6	Integration with Slice Manager		P2	
M2	Slice provision and deployment (preliminary) in the NFVI through SM		P2	
S.6.3	Automatic placement of a full NS through the Slice Manager		P2	
A7	Monitoring of services /virtual infrastructure		P3	
A10	Adaptation of Open5GCore benchmarking tool		P3	
A11	Adaptation of IXIA 5G evaluation tool with Open5GCore		P3	

4.3. Athens platform

4.3.1. MANO solution adopted and features

OSM release FIVE is adopted as the NFV MANO solution in Athens Platform. OSM is responsible for creating, modifying and deleting Network Services on the core and edge NFVI, as well as for day 0, day 1 and day 2 configuration. Some advanced functionalities, such as VNF placement, monitoring and day 1 configuration of the radio infrastructure components, will be performed by other components of the MANO layer, such as the Slice Manager, the EMS etc., as they are described in the respective D3.x.

Openstack is used for both core and edge VIM implementations. The core is comprised by a multi-node (three servers) Openstack Rocky release and an all-in-one Rocky release Openstack, while the edge is realized by two all-in-one and one multi-node (two servers) Openstack deployments in different regions of the Athens Platform.

For the realization of the 5G use cases that will be tested in Athens platform, a detailed description of which is included in D4.1, the above deployment will be used to host several NSs and VNFs:

- For the “Big Event” use case particular components of the AR application will be implemented either as VNFs or as MEC applications running on the available resources.
- For the second use case, “Eye in the sky”, two concurrent slices will be deployed, one eMBB for multimedia flows and one URLLC for navigation of the drone. Both slices will require the instantiation of several Network Services in both Core and Edge domains.
- The third use case requires the deployment of a Security as a Service (SeCaaS) in COSMOTE edge network.

Figure 18 depicts the deployment of the MANO layer components, as described in D4.1 [41]

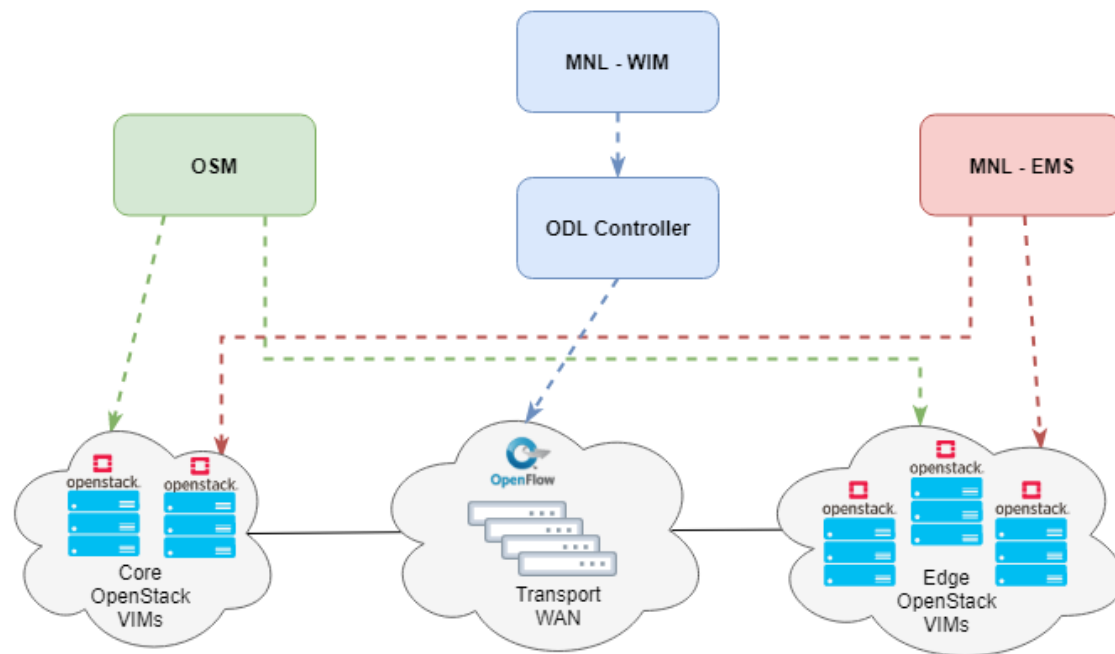


Figure 18 – Athens Platform MANO Deployment

4.3.2. Extensions developed as part of the Project

Due to the needs of the Athens platform regarding network connections, specific WIM and EMS, as part of the NMS, have been developed.

- **MNL WIM**

MNL-WIM (MediaNet Lab WIM) is implemented as a docker container running on a dedicated VM in Athens Platform. The MNL-WIM has a complete view of the Transport WAN that connects the core with the edge VIMs. It receives requests from the Slice Manager for the creation of a new network slice, creates the network graph based on the slice requirements and the available resources and then sends the requests for the creating the OpenFlow rules to the ODL controller. The communication between the WIM and the ODL is realized by REST API calls.

WIM is also responsible for the modification and the deletion of a created network graph. It uses an inventory for storing every element of the transport network, as well as the state and information of each created network graph.

- **MNL EMS**

MNL-EMS (MediaNet Lab EMS) is also implemented as a docker container running on a dedicated VM in Athens Platform. The initial release for the first phase of the project is used for the day 1 and day 2 configuration of the Amarisoft radio components [42], namely the vEPC


















deployed as VNF and the eNodeB deployed as PNF, that are currently used in the Athens platform.

4.3.3. Platform status (Release A)

Following the tasks list proposed in section 3.3.2, Table 8 represents the achievements accomplished by the Athens Platform, classified by phases.

As it can be seen, the tasks assigned for Phase 1 have already been fulfilled and most of the tasks scheduled for Phase 2, up to the submission of this deliverable.

Table 8 - Achievements Athens Platform

#	Task	Platform specific	status	
A.1	Deployment of NFVO	OSM R5	P1	
A.2	Deployment of core VIM	Openstack Rocky	P1	
S.2.1	Integration Core VIM - NFVO		P1	
S.2.2	Test NS deployment in cloud infrastructure (with manual placement)	vEPC	P1	
A.3	Deployment of edge VIM		P1	
S.3.1	Communication NFVO - core VIM - edge VIM		P1	
S.3.1.1	Deployment of two edge VIM at NCSRD	Openstack Rocky	P1	
S.3.1.2	Deployment of edge VIM at Cosmote	Openstack Rocky	P1	
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)		P1	
M.1	NS with manual placement deployed in both PoPs (Core and Edge)		P1	
A.4	WIM development and configuration for connecting VIM PoPs		P1	
S.4.1	SDN network configuration	ODL Version Oxygen	P1	
M.2	Manual deployment of an E2E 4G service	eMBB	P1	
A.5	Integration with Slice Manager		P2	
S.5.1	Slice provision (Core/Edge by Experiment template)		P2	
S.5.2	Deployment of a NS through the Slice Manager		P2	
M.3	Deployment of an E2E 4G service via the Slice Manager	eMBB	P2	

4.3.4. Future plan

Table 9 represents the subtasks that have not been accomplished yet and the phase each task is scheduled to be completed.

The deeper integration with the other components of the MANO layer, such as the Slice Manager and the Monitoring System, is essential for the realization of the functionalities that

Athens Platform will offer. Moreover, new releases of the WIM and the EMS are expected to support more components of the Infrastructure layers. Finally, an upgrade to the latest OSM release (Release SIX), which was published in 2019, will be examined.

Table 9 - Action plan Athens Platform

#	Task	Platform specific	status	
A.6	WIM development and configuration for connecting VIM PoPs		P2	
S.6.1	New Release of the WIM		P2	
S.6.2	Configuration of the SDN controller	Cosmote side	P2	
A.7	EMS development and new Release		P2	
S.7.1	New Release of the EMS to support the new 5G radio components		P2	
A.8	NSD/VNFD operations		P2	
S.8.1	VNFD/NSD onboarding		P2	
S.8.2	VNFD/NSD validation		P2	
S.8.3	Integration with the Portal		P2	
A.9	Integration with Slice Manager		P2	
S.9.1	Automatic placement of a full NS at VNF level through the Slice Manager		P2	
M.4	Deployment of an E2E 5G service via the Slice Manager		P2	
A.10	Monitoring of services /virtual infrastructure		P2	
S.10.1	Monitor individual VNF		P2	
S. 10.2	Monitor slice (aggregated monitoring)		P2	
S. 10.3	Alert received based on Analytics module (coming from T3.3)		P2	
S. 10.4	Endpoint NFVO configuration		P2	
S. 10.5	Rulebook- Policy engine (automation of actions based on analytics)		P3	
A.11	Upgrade NFVO version	OSM R6	P3	

4.4. Limassol platform

4.4.1. MANO solution adopted and features

The Limassol Platform is a new-born platform whose infrastructure and MANO components have been assembled from scratch creating a completely new infrastructure for experimentation. It has been designed to support use cases relating to maritime and rural/underserved areas based on satellite communications and with IoT LPWAN integration.

Thus, the use cases consider ubiquitous services with a specific latency and with strong point in the deployment time.

For that reason, virtualized Network Services are considered and need to be deployed and managed in a seamless manner. Moreover, to reduce the latency in some specific scenarios, some of these services will be deployed at the edge and orchestrated and managed to improve the use cases features and KPIs.

To achieve this objective, after a detailed study of the MANO solution depicted in section 2, the chosen architecture for the Limassol platform are a combination of OpenStack for the core, as well as the edge VIM, and OSM for the NFVO and VNF Manager.

Hence, the NFVI – VIM selected is OpenStack Rocky and the SDN controller will be ODL, the selected Management and Orchestration tool NFV MANO (NFVO) will be OSM Release Six to coordinate the resources and networks needed to set up cloud-based services and applications.

Satellite network connection

With respect to the satellite network segment, an iDirect Evolution satellite hub is employed at the Gateway and an iDirect X7 satellite router at the remote terminal. While the Evolution hub can only support 4G operation, 5G MANO of resources will be realized using an SDN switch.

It is worth noting that iDirect has recently produced the first release of 5G compliant software for its newer generation, Velocity, satellite hub that is capable of 5G MANO but unfortunately this newer system cannot be made available to the 5Genesis platform. This capability was developed in the EC H2020 project SaT5G; in this project a satellite compliant MANO system has been developed –described in deliverable D4.7 [43] – this development will be both validated and demonstrated at one of the three SaT5G testbeds.

Referring to the Limassol platform architecture Figure 19, 5G MANO will be implemented through the provision of separate VLANs between the SDN switch and the MEC host. Each VLAN will be configured to have different properties (e.g. IP address space, contention, bit rate, data volume, etc.). The SDN switch will map slices to VLANs thus implementing in essence network slicing over the satellite. To the best of our knowledge, this is a novel approach and could form the basis for several vertical use cases particularly allowing the provision of 5G satellite backhaul services over legacy systems.

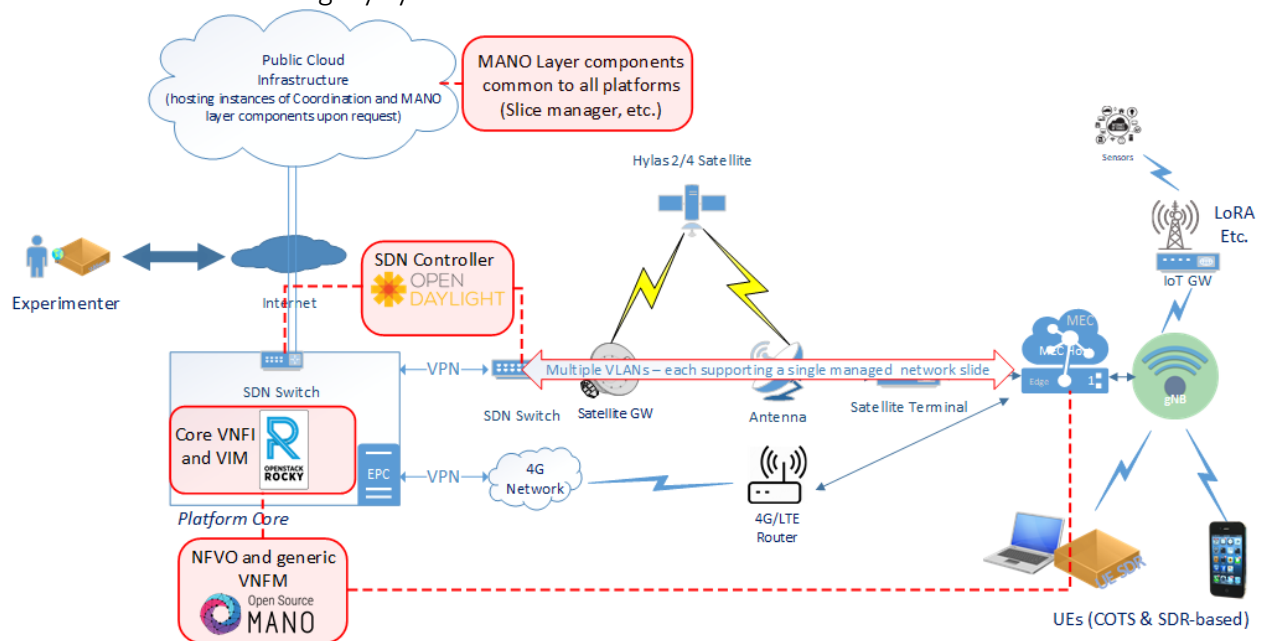


Figure 19 -- Limassol Platform MANO Deployment

4.4.2. Extensions developed as part of the Project

The following extensions are planned for the Limassol platform as part of the project:

- Extension and adaptation of OpenStack to also manage the satellite edge resources;
- Adaptation of the main platform-specific virtual network services (i.e. IoT vGW and link aggregation) so that their lifecycle can be managed by the MANO components;
- Integration of the MANO components with the ELCM and the Slice Manager for seamless experiment orchestration and execution;
- Integration of the satellite gateway management capabilities for end-to-end management of network resources.

Currently, specific VNF packages are being developed for the virtual network services oriented to the Use Cases. Hence, for the IoT vGW and link aggregation a new VNF will be included platform specific, apart from the common ones, transversal and shared by several platforms or slides. Moreover, for the integration with the Satellite Gateway a concrete approach in being done to achieve the orchestration and management also over the satellite communication as depicted in the previous subsection.

4.4.3. Platform status (Release A)

Currently, in the first cycle of implementation the deployment of the VIM core has been performed. OpenStack (Rocky release 18.1.0) has been installed at the platform core infrastructure at PLC premises and has been configured and tested to host future VNFs.









Within the core infrastructure the creation of VMs through the GUI provided by OpenStack has been done, in order to allocate and manage the resources to the virtual function and to access its state information.

The installation and configuration of the NFVO tool is being done. OSM release six as been installed and integrated with OpenStack and successfully connected to the VIM testing its functionalities.

Currently a basic topology network service (NS) has been deployed for testing purposes, including two VNFs with a single VDU (VM) instance each. These two VDUs are able to communicate through an internal network created from OSM as part of the deployed NS. In order to *ssh* into the created VDUs, the NS internal network was attached to a virtual router using Openstack and floating IP addresses were assigned to VDUs. The router is also attached to the existing management network to achieve this. An alternative to this implementation without using a router would be to attach VDUs to both networks (management and internal), but since general concept was to test OSM capabilities, floating IP functionality was included. Furthermore, initial integration of OSM with the ELCM (based on TAP) has been achieved. Currently a test plan has been implemented which interacts with OSM API. What this test plan achieves is to instantiate a NS, obtain the IP addresses of newly created VDUs and connect with *ssh* to one VDU to execute commands. Test steps and their corresponding instruments in TAP were developed as C# classes.

In the following table (Table 10), following the tasks listed in section 3.3.2, we observe the achievements accomplished by the Limassol Platform for this First Phase of integration within the MANO layer. Even though, not all tasks assigned for Phase 1 have been fulfilled (A2 almost fulfilled and A3 pending) and most of the tasks scheduled for Phase 2, up to the submission of this deliverable.













Table 10 – Achievements Limassol Platform

#	Task	Platform specific	status	
A.1	Deployment of NFVO	OSM	P1	
A.2	Deployment of core VIM	OpenStack Rocky	P1	
S.2.1	Integration Core VIM - NFVO		P1	
S.2.2	Test NS deployment in cloud infrastructure (with manual placement)		P1	
A.3	Deployment of edge VIM		P1	
S.3.1	Communication NFVO - core VIM - edge VIM		P1	
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)		P1	
M1	NS with manual placement deployed in both PoPs (Core and Edge)		P1	

4.4.4. Future plans

In the following table (Table 11) we can observe the sub-tasks that have not been yet accomplished as they belong to the future Phases of the task, so the is indicated the phase in which will be completed.

Table 11 - Action plan Limassol Platform

#	Task	Platform specific	status	
A.4	WIM development and configuration for connecting VIM PoPs	OpenDaylight	P2	
S.4.1	SDN network configuration	OpenDaylight	P2	
A.5	NSD/VNFD operations		P2	
S.5.1	VNFD/NSD onboarding		P2	
S.5.2	VNFD/NSD validation		P2	
S.5.3	Integration with the Portal		P2	
A.6	Integration with Slice Manager		P2	
S.6.3	Automatic placement of a full NS through the Slice Manager		P2	
A7	Monitoring of services /virtual infrastructure		P3	
S.7.2	Monitor slice (aggregated monitoring)		P3	
S.7.3	Alert received based on Analytics module (coming from T3.3)		TBD	
S.7.4	Endpoint NFVO configuration		TBD	

4.5. Surrey platform

4.5.1. MANO solution adopted and features

The architecture is modular by design, as it follows principles of local control of each test bed network by its provider as a separate administrative domain, whilst also providing access to the capabilities that each testbed offers to the 5GIC projects testbeds, by means of standard Application Programming Interfaces (API). As opposed to an approach which involves a thorough integration of multiple testbeds, this approach is highly scalable, since it allows future partners to join via the 5GIC-exchange without necessarily fully exposing their complex internal topology and internal networking solutions employed. It also promotes flexibility to prospective project partners, as the architecture does not impose requirements to support specific capabilities at partner testbeds.

The API between individual local testbed and the 5GIC Exchange uses Open Source MANO (OSM) which is compliant to the ETSI Management and Orchestration (MANO) standard. OSM is adopted in each local testbed. Through the northbound interface (NB I/F) of OSM, the 5GIC Exchange will be able to submit requests for Network Services (NS) to local testbeds. Such network services provide the network slices at testbed islands when the network service is requested by a user. A user may request services in a single testbed or across two different testbeds. Use cases for individual local testbeds will be presented later stage of this project, and end-to-end (E2E) use cases that involve two different local testbeds are communicating via the 5GIC Exchange.

Figure 20 below shows the high-level design of the multiple testbeds' connectivity via the 5GIC Exchange. In this diagram, the internal physical infrastructure at different local testbed may have different features and properties. Each testbed island runs a virtualisation layer, which is implemented using the virtualisation solutions chosen by the local testbed operator and orchestrated by a local OSM orchestrator. The 5GIC Exchange interacts with the OSM orchestrator of each local testbed.

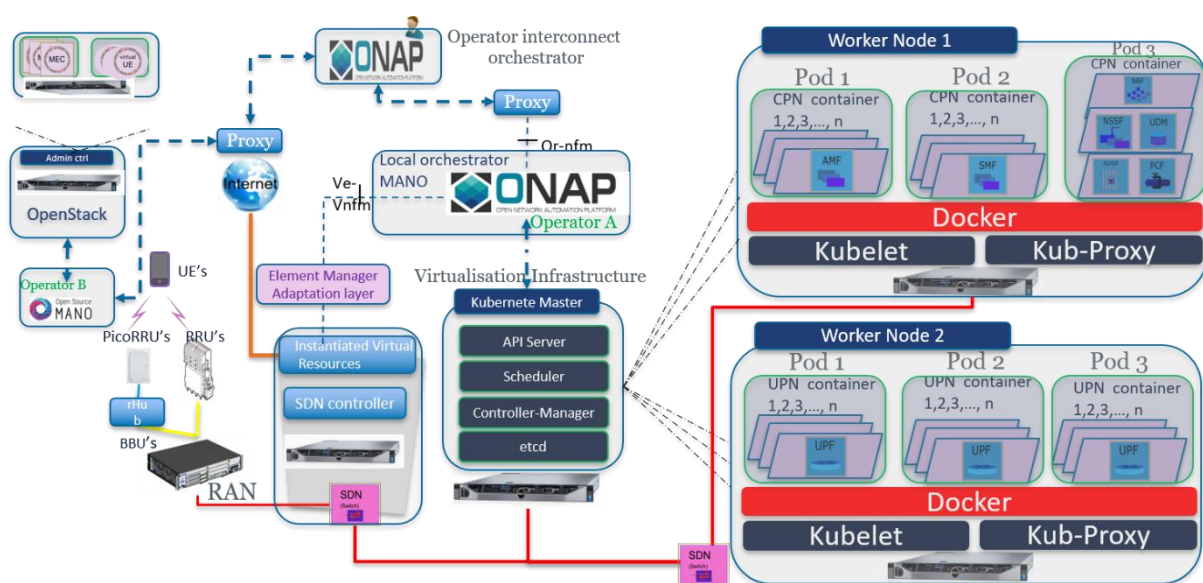


Figure 20 –Surrey Platform MANO Deployment

Figure 21 illustrates the implementation of business platform using TMF API's to allow business users to simply spin-up the new slices using their own business application.

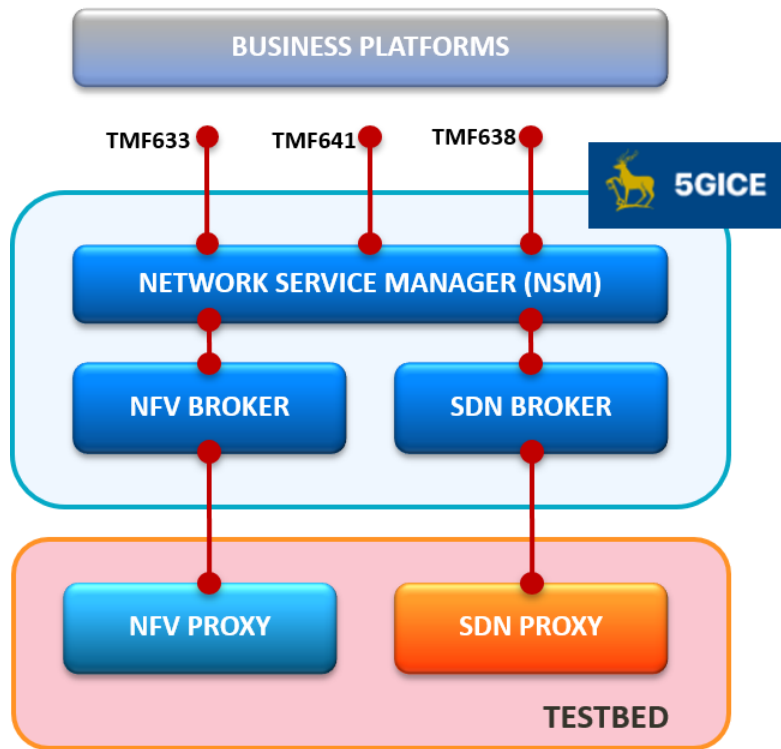


Figure 21 –Surrey Platform business platform setup using TMF API's

4.5.2. Policy Engine (APEX)

In the context of network automation for 5G, we also include in the management layer an option to optimise the slicing or other network components through the use of the APEX Adaptive Policy Execution engine. APEX, (as detailed in [45]) presents a strong tool for automated decision making, being able to handle adaptive policies, i.e. policies that can modify their behaviour based on system and network conditions, including decision making at runtime rather than at policy definition time and the ability to use context information that was not provided in the incoming event or request.

APEX is a versatile policy engine and can adopt various roles in the 5GENESIS project. APEX is an open-source policy engine that was developed by LMI and then released into the community as part of ONAP. However, the policy engine is not dependent on the other ONAP components and can be used to support dynamic decision making in any of our platforms. APEX is written in Java and runs on any platform that supports a JVM, e.g. Windows, Unix, Cygwin.

The APEX policy engine accepts input events and requests from other components, routes the input to the appropriate policies, computes the policy results, and generates response events or actions to be processed by other components. The policies may be affected by information injected into the policy context as changes in business or domain goals, by information derived from previous executions of the policies, and by context information retrieved from other components (analytics, inventory, topology, etc.).

An APEX engine has two main interfaces:

- An input interface to receive events: also known as ingress interface or consumer, receiving (consuming) events commonly named **triggers**, and
- An output interface to publish produced events: also known as egress interface or producer, sending (publishing) events commonly named **actions** or action events.

Two main components of APEX are the Trigger System that receives events which can trigger a policy and the Actioning System to send the result of a policy. The connections support various common technologies, such as messaging systems (Kafka, Web sockets), file input/output, and standard input/output. APEX supports context (as additional information) for all events as well as inside the engine to support policies. Context information can be read from any outside source.

In addition to triggers and actions, an APEX system will have different policies defined. A policy is defined in a Universal Execution Policy Specification (UEPS), directly executable in an APEX engine. Higher-level policy specifications (or existing policy languages) can be easily translated in UEPS.

An APEX system can use multiple policy engines with different policies deployed on each of them. Context information is automatically shared between all engine instances. A simple deployment component is provided.

APEX is best located close to the component whose behaviour it influences, i.e., if it is optimising the behaviour of a particular component in any of the layers, it should be located near that component. As a management and optimisation tool, we describe it in the context of the MANO layer.

Two examples that illustrate how APEX can be used for optimisations or repair actions, that we are investigating at the moment, are:

1. *Optimisation of slices based on dynamic conditions in the network*: In this case, the Analytics tool described in the Coordination layer will provide a predicted problem or a detected anomaly based on the monitored network data. This will trigger APEX to work with the Slice Manager component for slice update to minimize the impact of the potential problems coming from the network.

The following diagram depicts the flow of information and actions for the components involved in this use case:

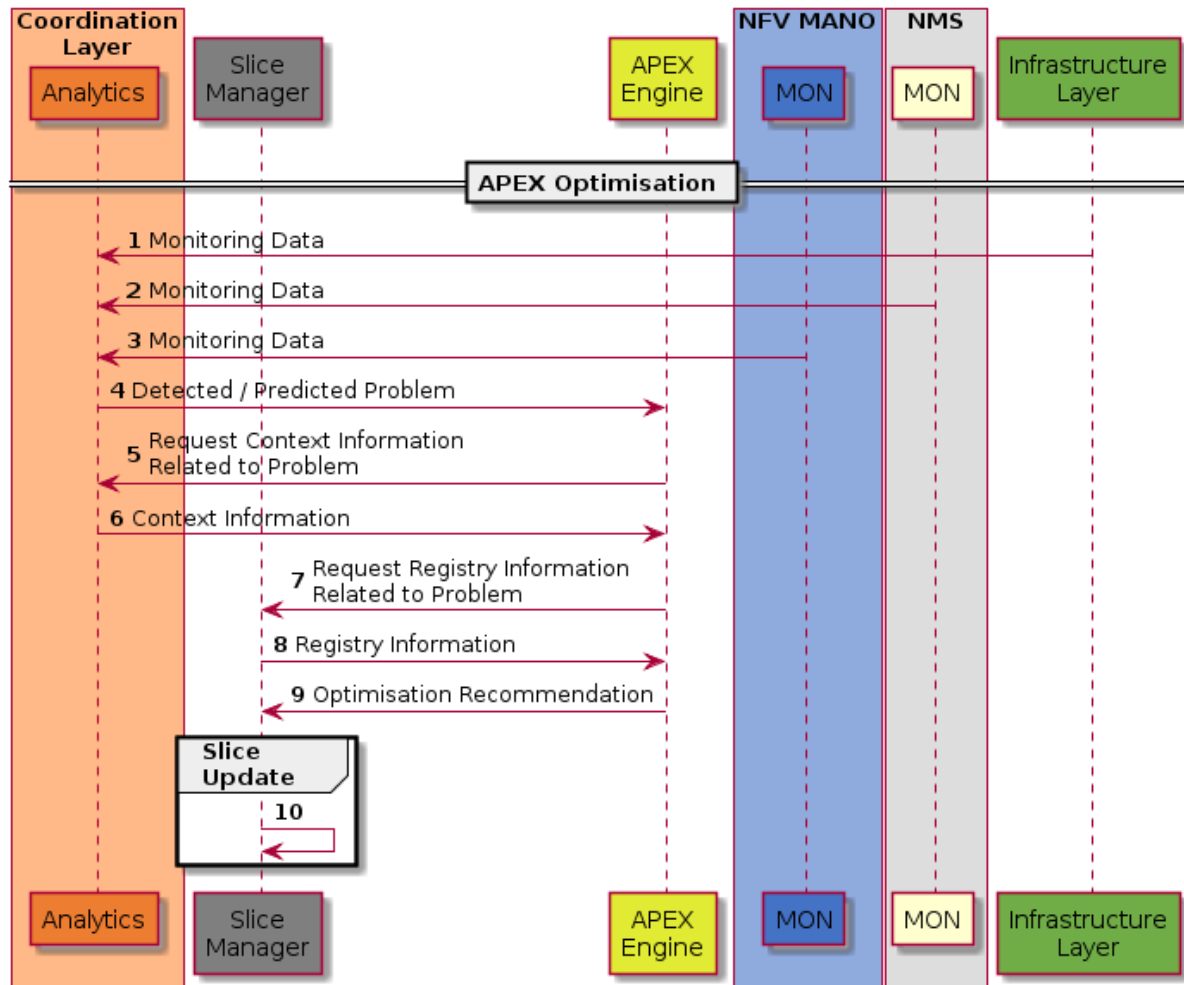


Figure 22 - Sequence diagram depicting the APEX use case of optimising slices based on dynamic conditions in the network.

The Analytics engine uses data from different monitoring components in the network (e.g., probes, monitoring blocks in NFV MANO & NMS components, etc.) and will send a trigger to APEX when a problem is predicted (or an anomaly shows that a problem is already present). This trigger will be analysed by APEX, who will also ask for context information (e.g., resources related to this problem) and possibly additional analytics data from the analytics engine. Once APEX has this data, it asks the Slice Manager component (in particular, the Registry) about the slice information stored in the registry, so that it can evaluate which slices are potentially impacted and take a decision that it can then send to the Slice Manager for a slice update information.

2. *Multi-domain slice repair:* 5GENESIS provides end-to-end slicing capabilities. The Slice Manager component is responsible for stitching together the individual domain slices. During the lifetime of a slice, if a problem is detected within one domain, but the local manager cannot detect the cause of the problem or fix the problem because it is related to inter-domain conditions or conditions stemming from a different domain than the one that observes the problem, APEX together with the Analytics component can help the Slice Manager understand how to fix the problem. The following diagram depicts the information flow in this case:

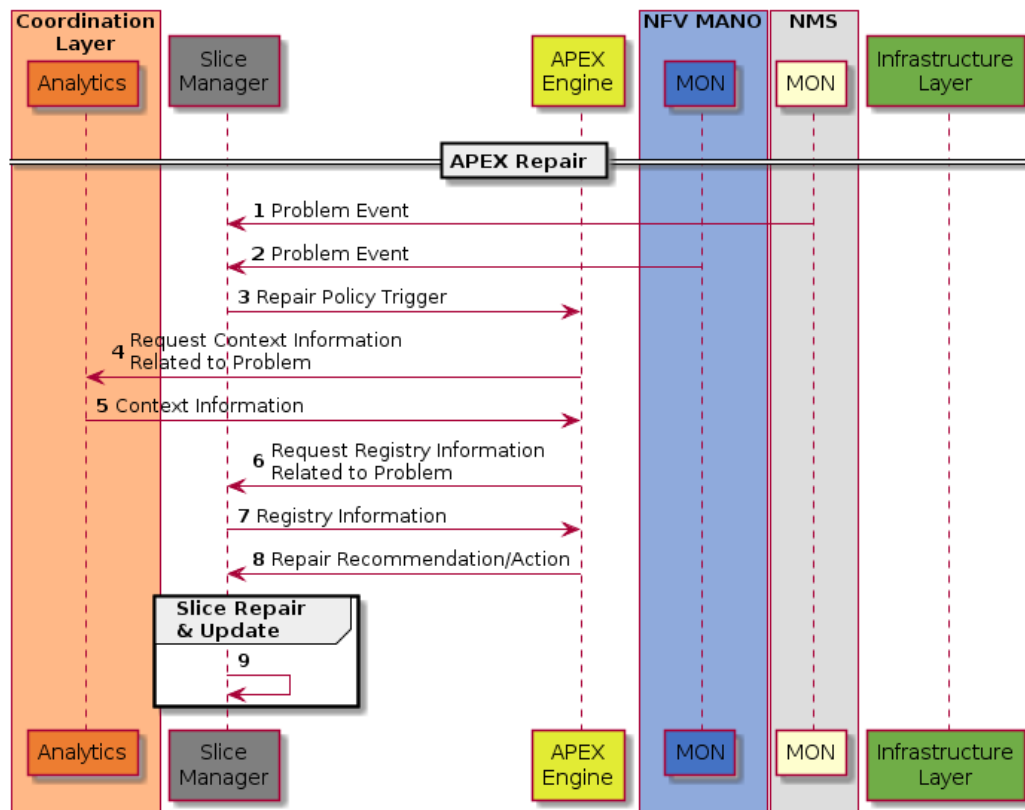


Figure 23 - Sequence diagram depicting the APEX use case of multi-domain slice repair.

As pictured above, individual domains will detect problems that they can't fix by themselves and send these up to the Slice Manager. The Slice Manager will then send a trigger to APEX to ask for help in deciding how to best solve this issue. APEX will ask the Analytics engine for additional context information and will also ask the Slice Manager for Registry information. Based on these, the right policy will be activated, and a repair action or recommendation will be sent back to the Slice Manager.

In our initial use cases, APEX works together with components in the MANO layer, as shown in Figure 24. However, the APEX component can be deployed and work in conjunction with any component that needs run-time dynamic decisions during the running of the experiments.

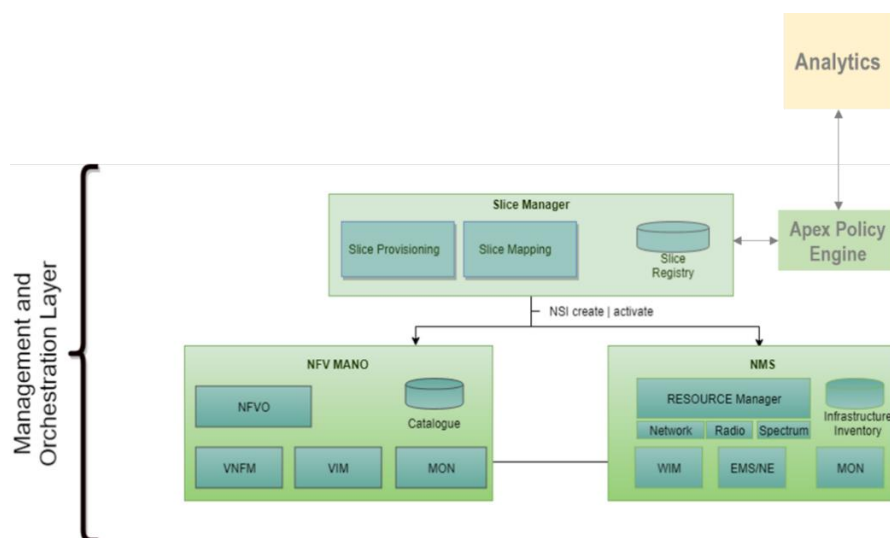






















Figure 24 - Location of the APEX Policy Engine within the MANO layer.

4.5.3. Platform status (Release A)

#	Task	Platform specific	status	
A.1	Deployment of NFVO	OSM R5	P1	
A.2	Deployment of core VIM	Openstack Queens	P1	
S.2.1	Integration Core VIM - NFVO		P1	
S.2.2	Test NS deployment in cloud infrastructure (with manual placement)		P1	
A.3	Deployment of edge VIM	Content caching MEC	P1	
S.3.1	Communication NFVO - core VIM - edge VIM		P1	
S.3.1.1	Edge SDN controller configuration	Ryu Customised SDN controller	P1	
S.3.2	Test NS deployment in Edge infrastructure (with manual placement)	Via OSM	P2	
M1	NS with manual placement deployed in both PoPs (Core and Edge)	Via OSM	P2	
A.5	NSD/VNFD operations	Via OSM	P2	
S.5.1	VNFD/NSD onboarding	Via OSM	P2	
A.6	Integration with Slice Manager		P2	
S.6.1	Slice provision (Core/Edge by Experiment template)		P2	
S.6.2	Deployment of a NS through the Slice Manager		P2	
M2	Slice provision and deployment (preliminary) in the NFVI through Slice Manager		P2	
A7	Monitoring of services /virtual infrastructure	Zabbix installed	P2	
S.7.1	Monitor individual VNF	Prometheus + Node exporter	P2	

4.5.4. Future plans

#	Task	Platform specific	status	
A.4	WIM development and configuration for connecting VIM PoPs		P2	
S.4.1	SDN network configuration	Ruy controller for SDN HW switches	P2	
A.5	NSD/VNFD operations		P2	

#	Task	Platform specific	status	
S.5.2	VNFD/NSD validation		P2	
S.5.3	Integration with the Portal		P2	
A.6	Integration with Slice Manager		P2	
S.6.3	Automatic placement of a full NS through the Slice Manager		P2	
A7	Monitoring of services /virtual infrastructure		P2	
S.7.2	Monitor slice (aggregated monitoring)		P2	
S.7.3	Alert received based on Analytics module (coming from T3.3)		P2	
S.7.4	Endpoint NFVO configuration		P2	
S.7.5	RuleBook- Policy engine (automation of actions based on analytics)		P3	

5. CONCLUSIONS

This document provides the initial version of the design and implementation of the Management and Orchestration of 5GENESIS, explaining the state of the art of different orchestrators and VIMs and analysing the adaptability to the 5GENESIS needs. Based on the 5GENESIS requirements described in section 3, the 5GENESIS architecture has been presented putting special attention on how it will be implemented in each platform and adapted in 5GENESIS. In this part, the common high-level requirements of the solutions have been identified:

- The selected NFVOs to be supported throughout 5GENESIS are OSM and Open Baton.
- The selected VIMs to manage the different NFVIs and Edge Infrastructures are OpenStack and OpenNebula.
- The NMS, even when it is a common component, requires a different implementation in each platform and using an existing framework for this task is not a valid option.

Finally, the current status and roadmap towards the next periods for each platform has been presented, and the results are described below:

- The chosen frameworks of the MANO components are presented as a good decision and the integration efforts are being carried out without mayor trouble.
- The testing of the individual components as well as the (partial) end-to-end is fulfilling the 5GENESIS expectations.
- Almost all platforms are on schedule, or even ahead of it, according to the initial task plan.

The result of this work will be updated on the next deliverable D3.2 at M30, once the final implementation and deployment will be completed.

REFERENCES

- [1] Rajendra Chayapathi, Syed F. Hassan, Paresh Shah, "Network Functions Virtualization (NFV) with a Touch of SDN", Nov 2016
- [2] European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV," ETSI GS NFV 003 - V1.2.1, Sophia Antipolis, 2014.
- [3] "What is NFV and what are its benefits" [Online], <https://www.networkworld.com/article/3253118/what-is-nfv-and-what-are-its-benefits.html>
- [4] "Network Functions Virtualisation (NFV) Release 3; Information Modeling; Report on External Touchpoints related to NFV Information Model" [Online], https://www.etsi.org/deliver/etsi_gr/NFV-IFA/001_099/024/03.02.01_60/gr_NFV-IFA024v030201p.pdf
- [5] 5GENESIS Consortium, "Deliverable D2.2 Overall Platform Design and Specifications", 2018
- [6] "Open Source MANO" [Online], <https://osm.etsi.org>
- [7] "Network Functions Virtualisation (NFV); Virtual Network Functions Architecture", https://www.etsi.org/deliver/etsi_gs/NFV-SWA/001_099/001/01.01.01_60/gs_NFV-SWA001v010101p.pdf
- [8] "Riftware" [Online], <https://riftio.com/riftware>
- [9] "Juju as a Service" [Online], <https://jaas.ai>
- [10] "OSM Release SIX" [Online], https://osm.etsi.org/wikipub/index.php/OSM_Release_SIX
- [11] Several authors, "OSM White Paper - OSM Release FIVE Technical Overview", Jan 2019 [Online], <https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseFIVE-FINAL.pdf>
- [12] "ETSI OSM Release SIX enhances Edge support and lets your Network Service fly" [Online], <https://www.etsi.org/newsroom/press-releases/1616-etsi-osm-release-six-enhances-edge-support-and-lets-your-network-service-fly>
- [13] "Open Baton" [Online], <https://openbaton.github.io>
- [14] "Network Functions Virtualisation (NFV); Management and Orchestration" [Online], https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf
- [15] "Topology and Orchestration Specification for Cloud Applications Version 1.0" [Online], <http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.pdf>
- [16] "Orchestra" [Online], <https://wiki.opnfv.org/display/PROJ/Orchestra>
- [17] "ONAP Platform Architecture" [Online]. <https://www.onap.org/architecture>
- [18] "Cloudify" [Online], <https://cloudify.co>
- [19] "ARIA TOSCA" [Online], <https://ariatosca.incubator.apache.org/>
- [20] "Orchestrating and managing VNFs on openstack" [Online], <https://www.slideshare.net/arthurberезin/orchestrating-and-managing-vnfss-on-openstack-demo-cloudify-openstack-fortigate-vrouter-etx>
- [21] "5Gtango" [Online], <https://5gtango.eu>
- [22] "OpenStack" [Online], <https://www.openstack.org>
- [23] "OpenNebula" [Online], <https://opennebula.org>

- [24] “Kubernetes” [Online], <https://kubernetes.io>
- [25] “Cloud Native Computing Foundation” [Online], <https://www.cncf.io>
- [26] 5GENESIS Consortium, “Deliverable D2.2 Overall Platform Design and Specifications”, 2018
- [27] 5GENESIS Consortium, “Deliverable D3.2 Slice Manager”, 2019
- [28] “What is Virtualized Infrastructure Manager (VIM)? Definition” [Online], <https://www.sdxcentral.com/nfv/definitions/virtualized-infrastructure-manager-vim-definition>
- [29] 5GENESIS Consortium, “Deliverable D2.1 Requirements of the Platform”, 2018
- [30] European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV); Management and Orchestration," ETSI GS NFV-MAN 001 V1.1.1, Sophia Antipolis, 2014.
- [31] European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV); Architectural Framework," ETSI GS NFV 002 V1.2.1, Sophia Antipolis, 2014
- [32] OSM Whitepaper Tech Content Release THREE, Section 3.5.1, [Online], <https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseTHREE-FINAL.pdf>
- [33] G. Gardikis et al., "An integrating framework for efficient NFV monitoring," *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, 2016
- [34] European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV); Management and Orchestration; VNF Packaging Specification," ETSI GS NFV-IFA 011 V2.1.1, Sophia Antipolis, 2016
- [35] “TOSCA Simple Profile for Network Functions Virtualization (NFV) Version 1.0” [Online], <https://docs.oasis-open.org/tosca/tosca-nfv/v1.0/tosca-nfv-v1.0.html>
- [36] European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV); Management and Orchestration; Network Service Templates Specification," ETSI GS NFV-IFA 014 V2.1.1, Sophia Antipolis, 2016
- [37] European Telecommunications Standards Institute, "Multi-access Edge Computing (MEC); Framework and Reference Architecture" ETSI GS MEC 003 – V2.1.1, Sophia Antipolis, 2019.
- [38] 5GENESIS Consortium, “Deliverable D4.4 The Malaga Platform – rel. A”, 2019
- [39] F. Luque-Schempp, L. Paniz, MM Gallardo, Pedro Merino-Gomez, “Optimizing the deployment of Virtual Network Functions in 5G networks with Model Based Testing”, *JCSD 2019, Zaragoza, 19-21 June 2019*
- [40] “Test Automation Platform (TAP)” [Online], <https://www.keysight.com/en/pc-2873415/test-automation-platform-tap?cc=ES&lc=eng>
- [41] 5GENESIS Consortium, “Deliverable D4.1 The Athens Platform – rel. A”, 2019
- [42] “Amarisoft” [Online], <https://www.amarisoft.com>
- [43] 5GENESIS Consortium, “Deliverable D4.7 The Limassol Platform – rel. A”, 2019
- [44] 5GENESIS Consortium, “Deliverable D4.13 The Berlin Platform – rel. A”, 2019
- [45] “Welcome to APEX - The Adaptive Policy EXecution (Engine)” [Online], <https://ericsson.github.io/apex-docs/>
- [46] “Policy” [Online], <https://wiki.onap.org/display/DW/Policy>

ANNEX 1 – VNF DESCRIPTORS

- OSM VNFD example

```
vnfd:vnfd-catalog:
  vnfd:
  - id: hackfest1-vnf
    name: hackfest1-vnf
    short-name: hackfest1-vnf
    version: '1.0'
    description: A simple VNF descriptor w/ one VDU
    logo: osm.png
    connection-point:
    - name: vnf-cp0
      type: VPORT
    vdu:
    - id: hackfest1VM
      name: hackfest1VM
      image: ubuntu1604
      count: '1'
      vm-flavor:
        vcpu-count: '1'
        memory-mb: '1024'
        storage-gb: '10'
      interface:
      - name: vdu-eth0
        type: EXTERNAL
        virtual-interface:
          type: VIRTIO
        external-connection-point-ref: vnf-cp0
    mgmt-interface:
      cp: vnf-cp0
```

- Open Baton VNFD example

```
{
  "vendor": "fokus",
  "version": "0.2",
  "name": "iperf-server",
  "type": "server",
  "endpoint": "generic",
  "configurations": {
    "name": "config_name",
    "configurationParameters": [
      {
        "confKey": "key",
        "value": "value"
      }
    ]
  },
  "vdu": [
    {
      "vm_image": [
        "ubuntu-14.04-server-cloudimg-amd64-disk1"
      ],
      "vimInstanceName": ["vim-instance"],
      "scale_in_out": 2,
      "vnfc": [
        {

```

```
        "connection_point":[
            {
                "floatingIp":"random",
                "virtual_link_reference":"private",
                "interfaceId":0
            }
        ]
    }
]
},
"virtual_link":[
    {
        "name":"private"
    }
],
"lifecycle_event":[
    {
        "event":"INSTANTIATE",
        "lifecycle_events":[
            "install.sh",
            "install-srv.sh"
        ]
    }
],
"deployment_flavour":[
    {
        "flavour_key":"m1.small"
    }
],
"vnfPackageLocation":"link_to_gitrepo",
"requires":{
    "server":{
        "parameters":["netname_floatingIp"]
    }
}
}
```

ANNEX 2 – NS DESCRIPTORS

- OSM NSD example: 2 VNFs

```
nsd:nsd-catalog:
  nsd:
  - id: hackfest2-ns
    name: hackfest2-ns
    short-name: hackfest2-ns
    description: NS with 2 VNFs connected by datanet and mgmtnet VLS
    version: '1.0'
    logo: osm.png
    constituent-vnfd:
    - vnfd-id-ref: hackfest2-vnf
      member-vnf-index: '1'
    - vnfd-id-ref: hackfest2-vnf
      member-vnf-index: '2'
    vld:
    - id: mgmtnet
      name: mgmtnet
      short-name: mgmtnet
      type: ELAN
      mgmt-network: 'true'
      vim-network-name: mgmt
      vnfd-connection-point-ref:
      - vnfd-id-ref: hackfest2-vnf
        member-vnf-index-ref: '1'
        vnfd-connection-point-ref: vnf-mgmt
      - vnfd-id-ref: hackfest2-vnf
        member-vnf-index-ref: '2'
        vnfd-connection-point-ref: vnf-mgmt
    - id: datanet
      name: datanet
      short-name: datanet
      type: ELAN
      vnfd-connection-point-ref:
      - vnfd-id-ref: hackfest2-vnf
        member-vnf-index-ref: '1'
        vnfd-connection-point-ref: vnf-data
      - vnfd-id-ref: hackfest2-vnf
        member-vnf-index-ref: '2'
        vnfd-connection-point-ref: vnf-data
```

- Open Baton NSD example: 1 VNF

```
{
  "name": "iperf-NSD",
  "vendor": "fokus",
  "version": "0.1-ALPHA",
  "vnfd": [ ... ],
  "vld": [
    {
      "name": "private"
    }
  ],
  "vnf_dependency": [
    {
      "source": {
        "name": "iperf-server"
```

```
    },  
    "target": {  
      "name": "iperf-client"  
    },  
    "parameters": [  
      "private"  
    ]  
  }  
]  
}
```

ANNEX 3 – ONAP POLICY ARCHITECTURE

As explained in [46], POLICY is a subsystem of ONAP that maintains, distributes, and operates on the set of rules that underlie ONAP's control, orchestration, and management functions. POLICY provides a logically centralized environment for the creation and management of policies, including conditional rules. This provides the capability to create and validate policies/rules, identify overlaps, resolve conflicts, and derive additional policies as needed. Policies are used to control, influence, and help ensure compliance with goals. Policies can support infrastructure, products and services, operation automation, and security. Users, including network and service designers, operations engineers, and security experts, can easily create, change, and manage policy rules from the POLICY Manager in the ONAP Portal. The figure below represents the target POLICY Architecture.

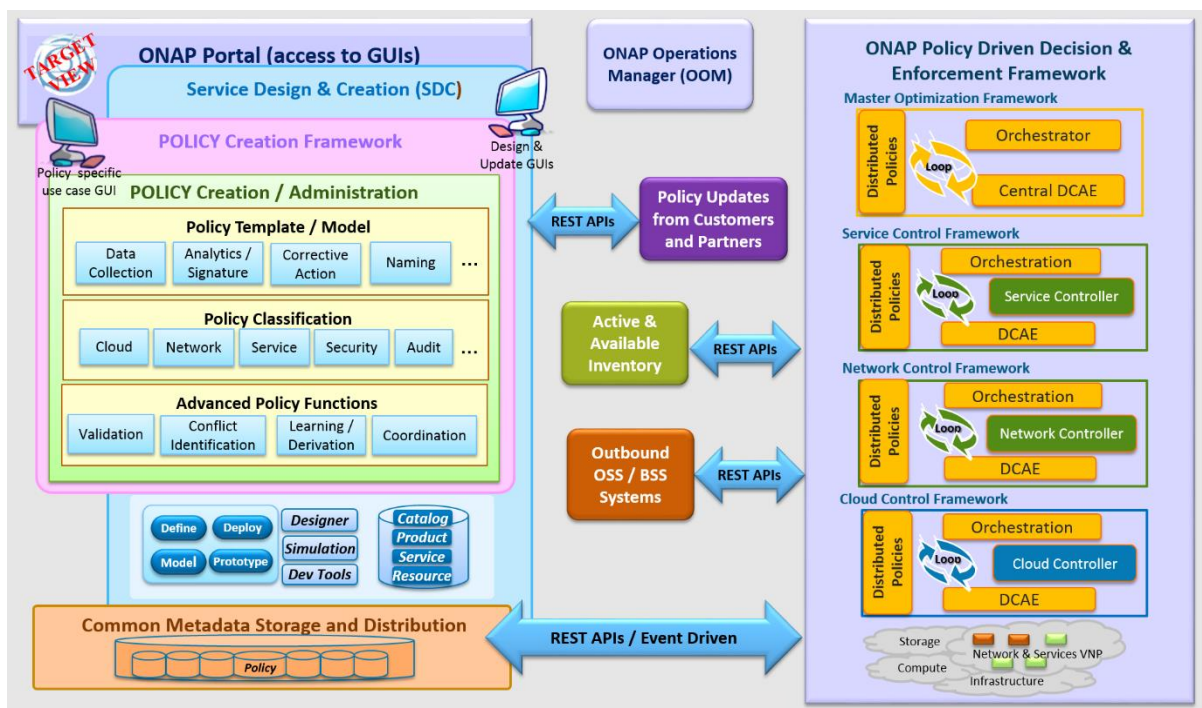


Figure 25 - ONAP target POLICY Architecture

The figure below represents the current POLICY Architecture.

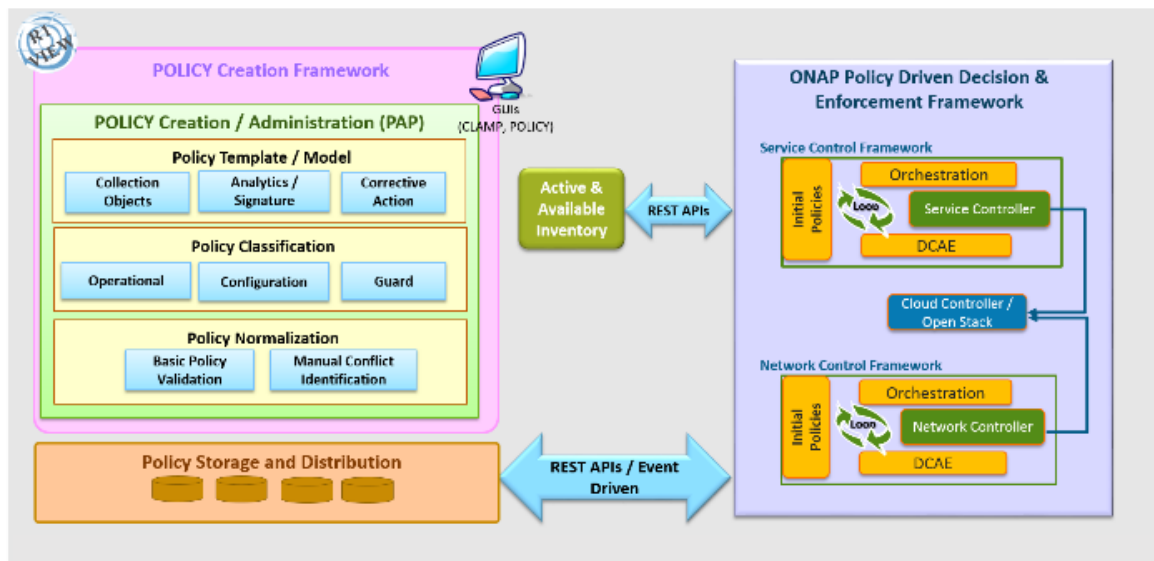


Figure 26 - ONAP current POLICY Architecture – Release 1

A policy is defined to create a condition, requirement, constraint, decision, or a need that must be provided, evaluated, maintained, and/or enforced. The policy is validated and corrected for any conflicts, and then placed in the appropriate repository, and made available for use by other subsystems and components. Alternately, some policies are directly distributed to policy decision engines such as Drools or XACML. In this manner, the constraints, decisions and actions to be taken are distributed.

ONAP POLICY is composed of several subcomponents: The *Policy Administration Point (PAP)*, which offers interfaces for policy creation, and two types of *Policy Decision Point (PDP)*, each based on a specific rules technology. PDP-X is based on XACML technology and PDP-D is based on Drools technology. PDP-X is *stateless* and can be deployed as a resource pool of PDP-X servers. The number of servers can be grown to increase both capacity (horizontal scalability) and to increase availability. The PDP-D is *stateful*, as it utilizes Drools in its native, stateful way and transactions persist so long as the PDP-D is active. Persistent Drools sessions, state management, local and geo-redundancy have been deactivated for the initial release of ONAP POLICY and can be turned on in a future release. Additional instances of XACML/Drools engines and assigned roles/purposes may also be added in the future to provide a flexible, expandable policy capability. As illustrated in the Figure below, the POLICY components are supported by a number of interfaces and subsystems. The ONAP portal provides a human interface for the creation, management and deployment of policies. It is a web-based system that utilizes POLICY APIs provided by the PAP.

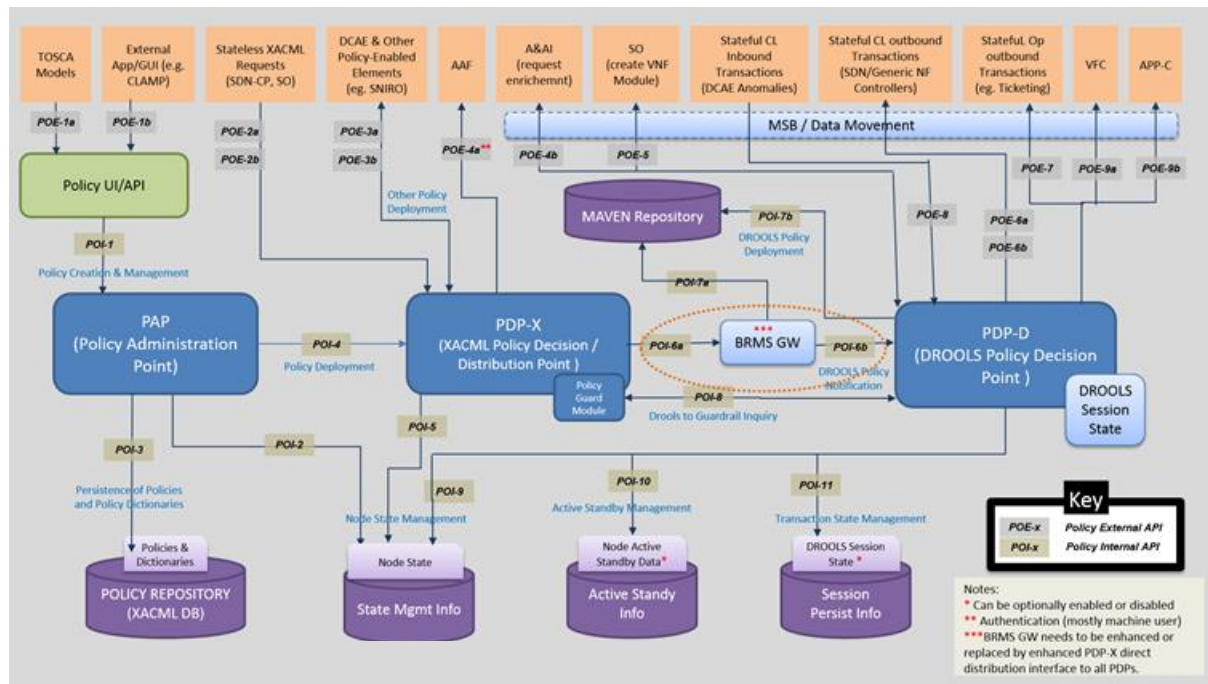


Figure 27 - ONAP POLICY creation & runtime Architecture (2018)

The PAP provides interfaces for the management of policies. It utilizes the XACML database to store policies, which are then distributed to the PDPs.

- Policy Creation

The Policy Creation component of the Policy subsystem enables creation of new policies and modification of existing policies, both during the design phase and during runtime. Policy Creation is targeted to be integrated to a unified Service Design and Creation (SDC) environment. A policy can be defined at a high level to create a condition, requirement, constraint, decision or a need that must be provided, evaluated, maintained, and/or enforced. A policy can also be defined at a lower or functional level, such as a machine-readable rule or software condition/assertion which enables actions to be taken based on a trigger or request, specific to particular selected conditions in effect at that time. Some examples of types of policies are:

- VNF placement — rules governing where VNFs should be placed, including affinity rules
- Data and feed management — what data to collect and when, retention periods, and when to send alarms about issues
- Access control — who (or what) can have access to which data
- Trigger conditions and actions — what conditions are actionable, and what to do under those conditions
- Interactions — how interactions between change management and fault/performance management are handled (e.g. should closed loops be disabled during maintenance)

- Policy Distribution

After a policy has been initially created or an existing policy has been modified, the Policy Distribution Framework sends the policy from the repository to its points of use, which include Policy Decision Points (PDPs) and Policy enforcement points (DCAE, Controllers, etc), before the policy is actually needed.

The decisions and actions taken by the policy are distributed. Policies are distributed either in conjunction with installation packages (for example, related to service instantiation) or independently, if unrelated to a particular service. Some policies can be configured (e.g., configuring policy parameters within microservices), while other policies are delivered to policy engines such as XAMCL and Drools. With this methodology, policies will already be available when needed by a component, minimizing real-time requests to a central policy engine or PDP (Policy Decision Point). This improves scalability and reduces latency.

- Policy Decision and Enforcement

Run-time policy enforcement is performed by ONAP subsystems that are policy-enabled or can respond to commands from a policy-enabled element such as a PDP. For example, policy rules for data collection are enforced by the data collection functionality of DCAE. Analytic policy rules, identification of anomalous or abnormal conditions, and publication of events signalling detection of such conditions are enforced by DCAE analytic applications. Policy rules for associated remedial actions, or for further diagnostics, are enforced by the correct component in a control loop such as the MSO, a Controller, or DCAE. Policy engines such as XACML and Drools also enforce policies and can trigger other components as a result (for example, causing a controller to take specific actions specified by the policy).