



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

[H2020 - Grant Agreement No. 815178]

Deliverable D3.3

Slice Management (Release A)

Editor T. Anagnostopoulos (NCSR)

Contributors NCSR (NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”), UMA (UNIVERSIDAD DE MALAGA), ATOS (ATOS SPAIN SA), INF (INFOLYSIS P.C.), INT (INTEL DEUTSCHLAND GMBH), IHP (INNOVATIONS FOR HIGH PERFORMANCE MICROELECTRONICS), LMI (L.M. ERICSSON LIMITED), FhG (FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V.), FON (FON TECHNOLOGY SL), UPV (UNIVERSITAT POLITÈCNICA DE VALÈNCIA)

Version 1.0

Date Oct 15th, 2019

Distribution PUBLIC (PU)



List of Authors

| | |
|--|---|
| NCSR | NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS” |
| T. Anagnostopoulos, G. Xilouris, H. Koumaras, T. Sarlas, S. Kolometsos, A. Gogos | |
| UMA | UNIVERSIDAD DE MALAGA |
| F. Luque, A. Diaz, B. Garcia | |
| ATOS | ATOS SPAIN SA |
| E. Jimeno, J. Melian | |
| INF | INFOLYSIS P.C. |
| V. Koumaras, A. Papaioannou | |
| INT | INTEL DEUTSCHLAND GMBH |
| V. Frasca | |
| IHP | INNOVATIONS FOR HIGH PERFORMANCE MICROELECTRONICS/LEIBNIZ-INSTITUT FUER INNOVATIVE MIKROELEKTRONIK |
| J. Gutiérrez | |
| LMI | L.M. ERICSSON LIMITED |
| A.-M. Bosneag | |
| FON | FON TECHNOLOGY SL |
| P. Salvador, L. Mallo, A. Pineda | |
| FhG | FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V. |
| M. Emmelmann | |
| UPV | UNIVERSITAT POLITECNICA DE VALENCIA |
| J. Suárez | |

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 D3.3 | T. Anagnostopoulos, Almudena Díaz | 15.10.2019 |

LIST OF ACRONYMS

| Acronym | Meaning |
|---------|--|
| 5G PPP | 5G Infrastructure Public Private Partnership |
| AP | Access Point |
| API | Application Programming Interface |
| AR | Augmented Reality |
| BSS | Business Support System |
| CEAP | Cloud-Enabled Application Platform |
| CESC | Cloud-Enabled Small Cell |
| CESCM | Cloud-Enabled Small Cell Manager |
| CLI | Command Line Interfaces |
| C-RAN | Cloud-RAN |
| CRUD | Create Read Update Delete |
| cSD-RAN | cloud Software Defined Radio Access Network |
| CSP | Content Service Provider |
| DB | Database |
| E2E | End to End |
| ELCM | Experiment Lifecycle Manager |
| 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 |
| EPC | Evolved Packet Core |
| ETSI | European Telecommunications Standards Institute |
| EU | European Union |
| gNB | gNodeB, 5G NR, next generation NR eq. of base station |
| GUI | Graphical User Interface |
| HNF | Hybrid Network Function |
| ICMP | Internet Control Message protocol |
| IoT | Internet of Things |
| KPI | Key Performance Indicator |
| LTE | Long-Term Evolution |
| M2M | Machine-to-Machine |
| MANO | MANagement and Orchestration |
| MEC | Mobile Edge Computing |
| mMTC | Massive Machine Type Communications-5G Generic Service |
| NBI | North Bound Interfaces |
| NF | Network Functions |
| NFV | Network Function Virtualisation |
| NFVI | Network Function Virtualisation Infrastructure |
| NFVO | Network Function Virtualization Orchestrator |
| NMS | Network Management System |
| NR | New Radio |
| NS | Network Service |

| Acronym | Meaning |
|---------|--|
| NSD | Network Service Descriptor |
| NSI | Network Slice Instance |
| NSMF | Network Slice Management Function |
| NSR | Network Service Record |
| NSSI | Network Slice Subnet Instance |
| NST | Network Slice Template |
| ODL | OpenDaylight SDN Controller |
| OF | OpenFlow |
| ONAP | Open Networking Automation Platform |
| OSM | Open Source MANO |
| OSS | Operations Support System |
| PCRF | Policy and Charging Rules Function |
| PDCP | Packet Data Convergence Protocol (PDCP) |
| PLMN | Public Land Mobile Network |
| PNF | Physical Network Functions |
| PoC | Proof of Concept |
| PoP | Point of Presence |
| QoS | Quality of Service |
| RAN | Radio Access Network |
| SBI | South Bound Interfaces |
| SDC | Service Design and Creation |
| SDK | Service Development Kit |
| SDN | Software Defined Network |
| SDR | Software Defined Radio |
| SLM | Slicing Lifecycle Manager |
| SM | Slice Manager |
| SO | Service Orchestration |
| SP | Service Platform |
| TAP | Test Automation Platform |
| TCP | Transmission Control Protocol |
| UDP | User Datagram Protocol |
| UE | User Equipment |
| uRLLC | Ultra-Reliable, Low-Latency Communications |
| VIM | Virtualised Infrastructure Manager |
| VLD | Virtual Link Descriptor |
| VNF | Virtual Network Function |
| VNFD | Virtual Network Function Descriptor |
| VNFR | Virtual Network Function Record |
| WIM | WAN Infrastructure Manager |
| WSMP | WIFI Service Management Platform |

Executive Summary

5GENESIS builds an experimentation facility that is accommodating experimenters to run validations and experiments on top of a 5G infrastructure. One of the prominent capabilities of the 5G technology is the ability to create network slices to satisfy the diverse requirements of each deployed vertical use case over the same infrastructure. This deliverable discusses and details the design, specifications and implementation of the 5GENESIS Network Slice Manager and more specifically the first release of the software. The slice manager is the component that manages the creation and provision of network slices over the infrastructure and is the binding element between the 5GENESIS Coordination layer and the actual infrastructure and management and orchestration layer, as shown in Figure 1. The Slice Manager receives the network slice template from the Coordination Layer and then provisions the slice, deploys the network services, configures all the physical and virtual elements of the slice and finally activates the end-to-end operation.

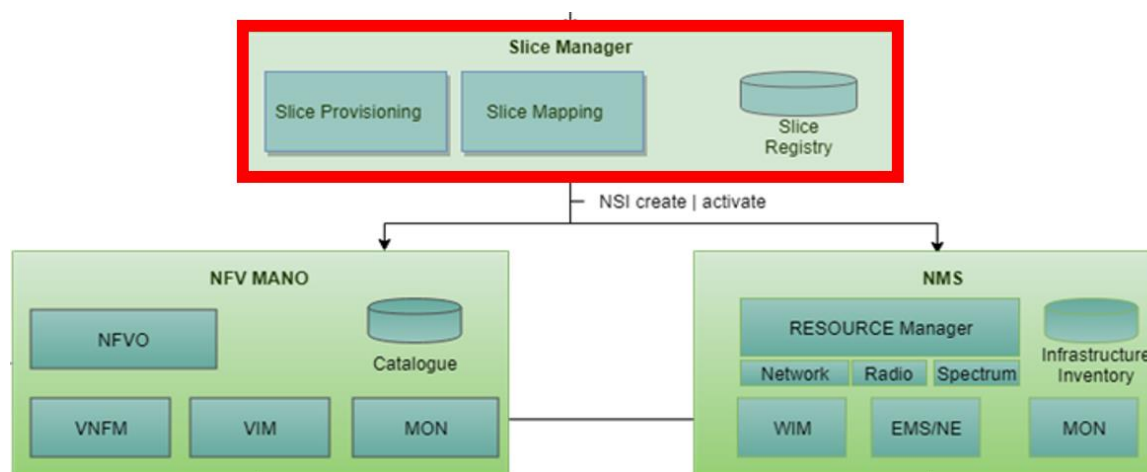


Figure 1 Slice manager component in the 5GENESIS architecture

Initially document analyses the main concepts and terminology of network slicing as well as the current status and progress related to slicing in various Standards organisations and opensource communities. In addition, implementations from relevant 5G European funded projects are analysed. The next part of the document, details analyses the requirements relevant to the Slice Manager and the system use cases. This section is vital for the architecture definition and the first release of the software. The Slice Manager architecture is exploiting microservices approach with all the communications between the components to be realised through a message bus. The design results in a high degree of modularity and expandability and can be easily enhanced with further improvements (i.e. high availability, health monitoring, new southbound drivers etc). The refined details of the lifecycle operations wrt the functional use cases of the previous section are also provided. Currently the supported lifecycle operations for network slices are creation, view, termination and modification. Furthermore, the functionalities and purpose of the northbound interfaces of the slice manager (NBI) are documented, providing information with respect to their implementation. The aforementioned interfaces are supporting REST APIs and swagger documentation is available along with the

software release. The document concludes with the Release A supported features and deployment guidelines and summarises on the future work and planned updates.

Although this document is meant to be self-reference and serve as a complete documentation for the slice manager, readers are encouraged to refer also to the WP3 deliverables released in parallel and more specifically the one on **Management and orchestration (Release A) (D3.1)**.

Table of Contents

| | |
|--|-----------|
| LIST OF ACRONYMS | 6 |
| 1. INTRODUCTION..... | 13 |
| 1.1. Purpose of the Document..... | 13 |
| 1.1.1. Document Dependencies | 13 |
| 1.2. Structure of the Document..... | 14 |
| 1.3. Target Audience | 14 |
| 2. CONCEPTS AND RELATED WORK..... | 15 |
| 2.1. Network Slicing Concepts | 15 |
| 2.2. Standardization Related Activities..... | 15 |
| 2.2.1. OSM | 15 |
| 2.2.2. ONAP | 17 |
| 2.3. Project Based Implementations | 18 |
| 2.3.1. 5GTANGO | 18 |
| 2.3.2. MATILDA..... | 19 |
| 2.3.3. 5GESSENSE – EmPOWER | 20 |
| 3. SLICE MANAGER SPECIFICATIONS..... | 22 |
| 3.1. Relation with overall 5GENESIS requirements | 22 |
| 3.2. Use Cases..... | 24 |
| 3.2.1. Network Slice Instance Creation | 24 |
| 3.2.2. Network Slice Subnet Instance Creation | 24 |
| 3.2.3. Network Slice Instance Termination | 25 |
| 3.2.4. Network Slice Subnet Instance Termination | 25 |
| 3.2.5. Network Slice Instance Modification | 25 |
| 3.2.6. Network Slice Subnet Modification | 26 |
| 4. 5GENESIS SLICE MANAGER | 27 |
| 4.1. Overview | 27 |
| 4.2. Slice Manager Architecture | 28 |
| 4.2.1. Components..... | 29 |
| 4.2.1.1. North Bound Interface API | 29 |

| | |
|--|-----------|
| (a) CLI Tool | 29 |
| (b) GUI | 30 |
| 4.2.1.2. Slicing Lifecycle Manager | 30 |
| 4.2.1.3. Slice Mapping | 30 |
| 4.2.1.4. Slice Provisioning | 31 |
| 4.2.1.5. Slice Monitoring | 32 |
| 4.2.1.6. Adaptation Layer | 32 |
| 4.2.1.7. Databases | 32 |
| (a) Infrastructure Repository | 32 |
| (b) NSI Record Repository | 32 |
| (c) Supported E2E Service Catalogue | 33 |
| 4.2.1.8. Message Queueing software | 33 |
| 4.3. Slice Lifecycle Operations | 33 |
| 4.3.1. Slice Creation | 33 |
| 4.3.2. Slice Read | 37 |
| 4.3.3. Slice Update | 37 |
| 4.3.4. Slice Deletion | 41 |
| 4.4. Registered End-to-End Services | 42 |
| 5. INTERFACES | 43 |
| 5.1. Slice Manager Interfaces | 43 |
| 5.1.1. Elcm-Sm | 43 |
| 5.1.2. Sm-Vim | 44 |
| 5.1.3. Sm-Nfvo | 44 |
| 5.1.4. Sm-Wim | 44 |
| 5.1.5. Sm-Ems | 44 |
| 5.1.6. Sm-Mon | 45 |
| 5.2. Integration with the other 5GENESIS Components | 45 |
| 6. RELEASE A OF 5GENESIS SLICE MANAGER | 46 |
| 6.1. Supported features and functionalities | 46 |
| 6.1.1. Supported features and functionalities | 46 |
| 6.1.2. Supported MANO layer components | 50 |
| 6.1.3. Network Slice Template | 50 |
| 6.2. Integration and deployment | 50 |
| 6.2.1. Deployment and integration at Athens platform | 50 |
| 6.2.2. Deployment and integration at Malaga platform | 51 |

| | |
|---|-----------|
| 6.3. Preliminary Testing | 52 |
| 6.3.1. Proof of Concept | 52 |
| 6.3.2. Slice Creation Time | 53 |
| 7. RELEASE A SUMMARY AND FUTURE PLANS..... | 55 |
| 8. REFERENCES | 57 |
| 9. APPENDICES..... | 59 |
| 9.1. North Bound Interface REST APIs..... | 59 |
| 9.1.1. Network Slice | 59 |
| 9.1.1.1. Endpoints | 59 |
| 9.1.1.2. Examples | 59 |
| 9.1.2. NFVO..... | 60 |
| 9.1.2.1. Endpoints | 60 |
| 9.1.2.2. Examples | 60 |
| 9.1.3. WIM | 61 |
| 9.1.3.1. Endpoints | 61 |
| 9.1.3.2. Examples | 62 |
| 9.1.4. VIM | 62 |
| 9.1.4.1. Endpoints | 62 |
| 9.1.4.2. Examples | 63 |
| 9.1.5. EMS..... | 63 |
| 9.1.5.1. Endpoints | 63 |
| 9.1.5.2. Examples | 64 |
| 9.1.6. Registered End-to-End Services | 64 |
| 9.1.6.1. Endpoints | 64 |
| 9.1.6.2. Examples | 65 |
| 9.2. MANO Component Registration Templates | 66 |
| 9.2.1. New VIM Template | 66 |
| 9.2.2. New NFVO Template | 66 |
| 9.2.3. New WIM Template..... | 67 |
| 9.2.4. New EMS Template | 67 |
| 9.2.5. New Monitoring System Template | 67 |
| 9.3. Versioning Policy | 67 |
| 9.4. Network Slice Template | 68 |

1. INTRODUCTION

1.1. Purpose of the Document

Wireless communication growth has been on the rise, reaching newer industry segments such as the automotive and health industries. Each segment imposes different requirements upon the communications systems such as ultra-high bandwidth or low latency, for example. In this context applications such as machine-to-machine M2M communications, Internet of Things (IoT), multimedia and Augmented Reality (AR) are requiring communications systems that can perform dynamic resource utilization. Network slicing will play a pivotal role in addressing varied use cases by enabling dedicated virtualized network slices for each use case. Network slicing is about transforming a PLMN from a single network to a network where logical partitions are created, with appropriate network isolation, resources, optimized topology and specific configuration to serve various service requirements.

Slice Manager creates multiple dedicated virtual networks using a common physical infrastructure. Each virtual network slice is composed of independent logical network functions serving a specific use case. Each network slice will be optimized to provide the required resources and quality of service (QoS) with regards to latency, throughput, capacity, coverage and so on.

This document discusses the specifications and Release A implementation of the 5GENESIS Slice Manager. The Slice Manager developed in the frame of 5GENESIS is responsible for communicating southbound with the management and control elements of the infrastructure and northbound with the 5GENESIS Coordination Layer components. Although 5GENESIS Slice Manager is mostly aiming in supporting the experimental features of the 5GENESIS facilities, it will also in the next releases implement additional features related to the network slice provisioning and service deployment. The Slice Manager is exploiting enabling technologies available at the infrastructure layer in order to be able to satisfy the aforementioned requirements.

This document presents the detailed architecture of the Slice Manager and the functional Use Cases that it supports and concern the lifecycle management of the network slices. It discusses the overall architecture and provides information on each functional component of the architecture. The architecture of the Slice Manager is exploiting to design principles namely a) microservices and b) modularity. In this sense, the design allows implementation of additional interfaces or features without deteriorating system operation. The document provides detailed information on the available API for the northbound interfaces and implements interfaces with well-known MANO and NMS solutions.

1.1.1. Document Dependencies

This document is based on specifications, requirements and assumptions as discussed in the first release of the Architecture related deliverables. In addition, there is a strong reliance on the actual infrastructure elements that may be controlled and managed by the Slice Manager and depend on each platform as presented in the deliverables of WP4. The table below summarizes the relevance towards the deliverables produced by WP2.

Table 1 Document dependencies

| id | Document title | Relevance |
|----------|---|--|
| D2.1 [1] | Requirements of the Facility | 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 [2] | 5GENESIS Overall Facility Design and Specifications | The 5GENESIS facility architecture is defined in this document. The list of functional components to be deployed in each testbed is defined. |
| D2.3 [3] | Initial planning of tests and experimentation | Testing and experimentation specifications that influence the testbed definition, operation and maintenance are defined. |

1.2. Structure of the Document

The document presents in the second chapter the network slicing concept as defined by the relevant SDOs (i.e. 3GPP) and attempts a brief literature survey, updating the one conducted for D2.1 [1]. Chapter 3 provides an overview of the Slice Manager specifications and its functional use cases. Chapter 4 discusses in detail the architecture of the Slice Manager. Details on the specification of interfaces are provided in Chapter 5. Finally, at Chapter 6 the Release A of the Slice Manager is presented. Conclusions chapter wraps up the document and provides insights on the planned future work.

1.3. Target Audience

This document besides providing a relative well-rounded information regarding the slice manager that is being implemented in 5GENESIS is also providing information and documentation regarding implemented and future features and capabilities. In this context the document is targeted to developers and researchers that work on similar network slicing concepts and/or on management and coordination systems that could use the open APIs provided by the Slice Manager in order to interface with the 5GENESIS infrastructure. Finally, since the software code base is open source, this document serves also as an introduction and architecture primer of the 5GENESIS Slice Manager.

2. CONCEPTS AND RELATED WORK

2.1. Network Slicing Concepts

Based on the 3GPP Specification 28.801 “Telecommunication management; Study on management and orchestration of network slicing for next generation network” [4], the following concepts have been defined:

- Network Slice Instance (NSI): An NSI includes all functionalities and resources necessary to support certain set of end-to-end communication services.
- Network Slice Template (NST): The NST describes the Network Slice to be created.
- Components of an NSI: The NSI is comprised of Virtual or Physical Network Functions (NFs). These NFs can be dedicated to an NSI or shared among multiple NSIs. If the NFs are interconnected, the Slice Manager contains information relevant to connections between these NFs, such as topology of connections, network graph, link requirements, etc. This information is either loaded to the Slice Manager using the Supported End-to-End Services described in 4.4 or included in the NST.
- Network Slice Subnet Instance (NSSI): An NSI may be composed of NSSIs. The NSSIs may represent different domains of the physical infrastructure, such as the NFVI, Transport Network, RAN, etc. A NSSI may include other NSSI(s).

2.2. Standardization Related Activities

2.2.1. OSM

In its release FIVE, ETSI OSM [5] introduced major enhancements to support network slicing features. OSM has integrated slice manager and extended information model to allocate network slice template (NST) and network slice instance (NSI).

OSM has a common information model for defining the networking part of the network functions (either VNFs, PNFs or hybrid ones) and network services, which eases the complexity of the network operations and drastically simplifies and automates daily operations. OSM network slice feature takes advantage of that also with its IM, allowing network services to stay self-contained and agnostic to technology infrastructure for completely different network characteristics in each service.

Following the NSD approach, each slice is modelled as a set of network services connected by networks or VLDs. Figure 2 shows an example of how two NSs are interconnected with virtual links to build a slice and Table 2 contains a simple example of the NST model represented in the figure.

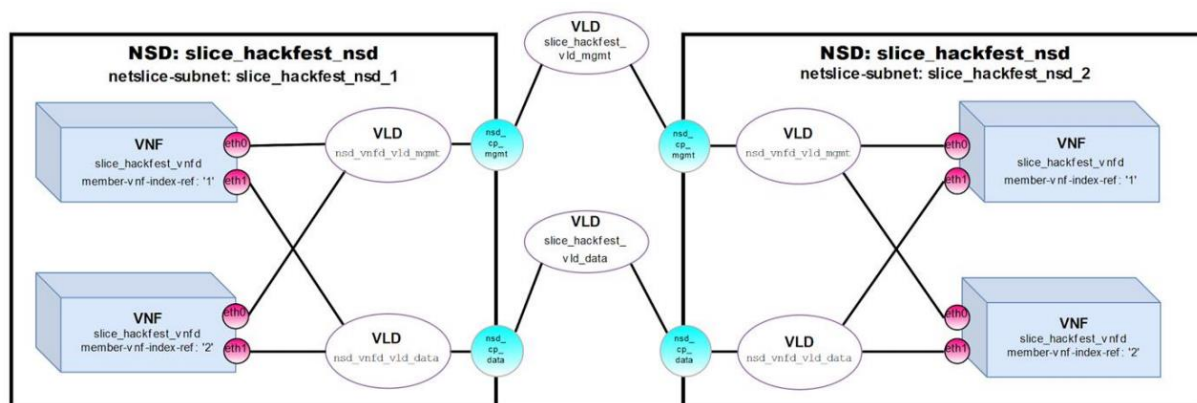


Figure 2: NST diagram example

OSM slicing allows shared network services, which not only helps to overcome the problem of fixed devices (that can be represented in OSM as PNFs) within the network that cannot be moved and need to be used by different actors at the same time, but also introduces flexibility when using regular VNFs.

Considerations:

- Network Slice Subnet can be considered as an NFV Network Service.
- Network Functions can be described as VNF and PNF.
- If the shared NS has already been deployed, it will not be re-deployed, but it will be reused: the initial configuration for the second network slice can still be done in the shared NS to let it know that new elements are present.

Table 2: Network Slice Template

```
nst:
- id: slice_hackfest_nst
  name: slice_hackfest_nst
  SNSSAI-identifier:
    slice-service-type: eMBB
  quality-of-service:
    id: 1
netslice-subnet:
- id: slice_hackfest_nsd_1
  is-shared-nss: 'false'
  description: NetSlice Subnet (service) composed by 2 vnfs and 4 cp (2 mgmt and 2 data)
  nsd-ref: slice_hackfest_nsd
- id: slice_hackfest_nsd_2
  is-shared-nss: 'false'
  description: NetSlice Subnet (service) composed by 2 vnfs and 4 cp (2 mgmt and 2 data)
  nsd-ref: slice_hackfest_nsd
netslice-vld:
- id: slice_hackfest_vld_mgmt
  name: slice_hackfest_vld_mgmt
  type: ELAN
  mgmt-network: 'true'
  nss-connection-point-ref:
    - nss-ref: slice_hackfest_nsd_1
      nsd-connection-point-ref: nsd_cp_mgmt
    - nss-ref: slice_hackfest_nsd_2
      nsd-connection-point-ref: nsd_cp_mgmt
- id: slice_hackfest_vld_data
  name: slice_hackfest_vld_data
  type: ELAN
  nss-connection-point-ref:
    - nss-ref: slice_hackfest_nsd_1
      nsd-connection-point-ref: nsd_cp_data
    - nss-ref: slice_hackfest_nsd_2
      nsd-connection-point-ref: nsd_cp_data
```


The interoperation with the slice in OSM is done by running day-two primitives (Juju), that are mapped to a sequence of calls to NS primitives, which, in turn, are derived in calls to VNF primitives.

2.2.2. ONAP

Since ONAP Casablanca release [6], there was a strong focus of ONAP use cases towards the support and implementation of 5G proof of concepts (PoCs). In fact, Casablanca release specifies the following use cases towards 5G:

- Lifecycle management of physical network functions (PNFs), critical for the appropriate orchestration of physical elements that are mostly deployed at the edge or the backhaul of the 5G infrastructure.
- TOSCA representation of PNFs (i.e. model, syntax and semantics)
- Real-time performance monitoring and data collection

Effectively the efforts related to the first two use cases concentrated at implementing specific enhancements to deploy Edge PNF and Virtual Radio Network Functions and in addition the TOSCA representation attempted to define the mapping between the PNF resources used in ONAP services and the physical devices themselves.

The work above is continued in the next release of ONAP, namely Dublin. The work focuses in the following aspects (mentioning only the ones relevant to 5G):

- Modelling of 5G network slicing
- Enhancing ONAP support for NETCONF for 5G use cases
- Registration for PNF plug n'play registration
- Enabling ONAP to manage Edge resource and VNFs deployment
- E2E life-cycle management and assurance of Broadband Services
- Virtualized and Containerized networking workloads deployment in Kubernetes based Cloud regions via MultiCloud/k8s plugin

Currently, a proposed modelling high level example was demonstrated on ONAP committees F2F modelling meeting. It is based on current modelling capabilities in Service Design and Creation (SDC). This example model can be used as a trigger for requirements to support Network Slicing in SO, A&AI, CCSDK, CDS, OOF, VID, MultiCloud, Policy, DCAE. The shared services concept that is needed for Network Slicing is still discussed in Modelling subcommittees

Given the vast code base of ONAP, it is expected that concrete changes will be available with EI-Alto release next year (2020). The work strongly depends on availability of vRAN/RAN simulators and 5G Core components and VNFs/simulators. Plus, it expected that modelling of transport between RAN and Core for the network slice subnets to be concluded.

2.3. Project Based Implementations

2.3.1. 5GTANGO

5GTANGO [7] is a 5GPPP Phase2 Innovation Action that enables the flexible programmability of 5G networks with: a) NFV enabled Service Development Kit(SDK), b) a validation and verification platform (VnV) for VNFs/Network services qualification and c) a modular Service Platform, the SONATA SP. SONATA SP consists of a number of software modules which together provide the required functionality for network service creation, deployment, and management.

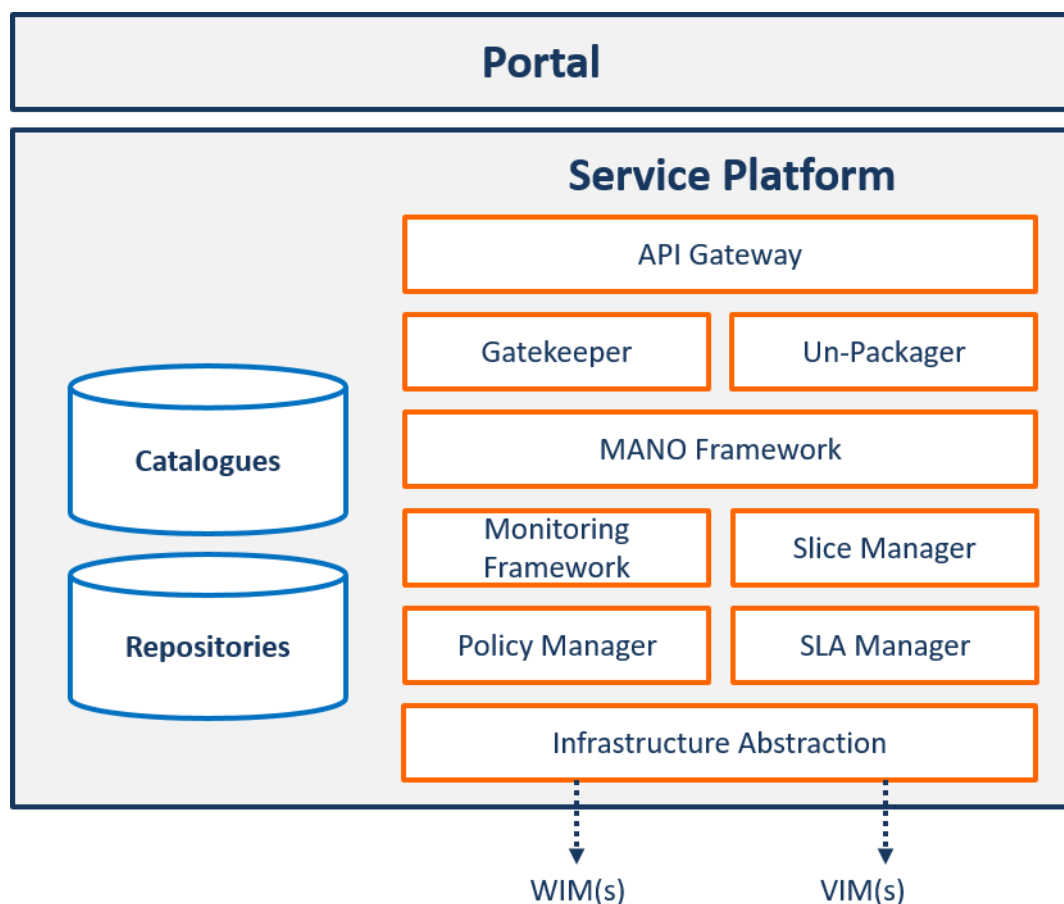


Figure 3: SP Architecture

In the later SONATA releases a Network Slice Manager [8] has been added. The Slice Manager deals with two objects. The Network Slice Template descriptor (NSTd), which defines a set of network services to be instantiated, and the Network Slice Instantiation record (NSIr), which manages the information of the deployed network services.

There are two main features in the NSM:

- Network Services Composition: the ability to compose multiple network services among them in order to compose a complete and more isolated network service to the final user
- Network Service Sharing: it allows the sharing of a Network Service deployment among multiple network slices so that they can be reused and be resource efficient

2.3.2. MATILDA

MATILDA [9], a H2020-ICT-2016-2 Innovation Action project, introduced the extended Operations and Business Support Systems (OSS/BSS) as their solution for the Slice Manager [10]. As depicted in Figure 4, the extended OSS/BSS, alongside the Computing Slice Manager and the NFVO, comprise the network platform, a middle layer that is responsible for providing the required end-to-end network services to connect the application components deployed in diverse datacentres among themselves, and with the mobile terminals (i.e., 5G User Equipment – UE) consuming and/or providing data to the vertical application.

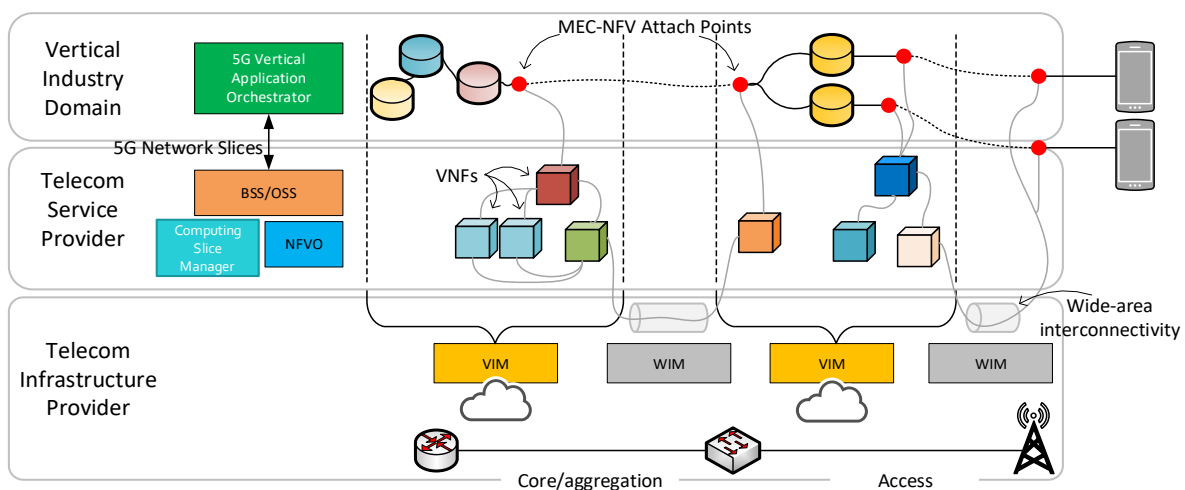


Figure 4: MATILDA High-level System Architecture

The extended OSS/BSS is in charge of setting up, monitoring and managing the lifecycle of any network services and exposing them to vertical applications. It has been designed according to a highly modular architecture, composed of a mesh of microservices, in charge of managing the different components of the whole 5G infrastructure, as well as of the overlying network and applicative services, as shown in Figure 5.

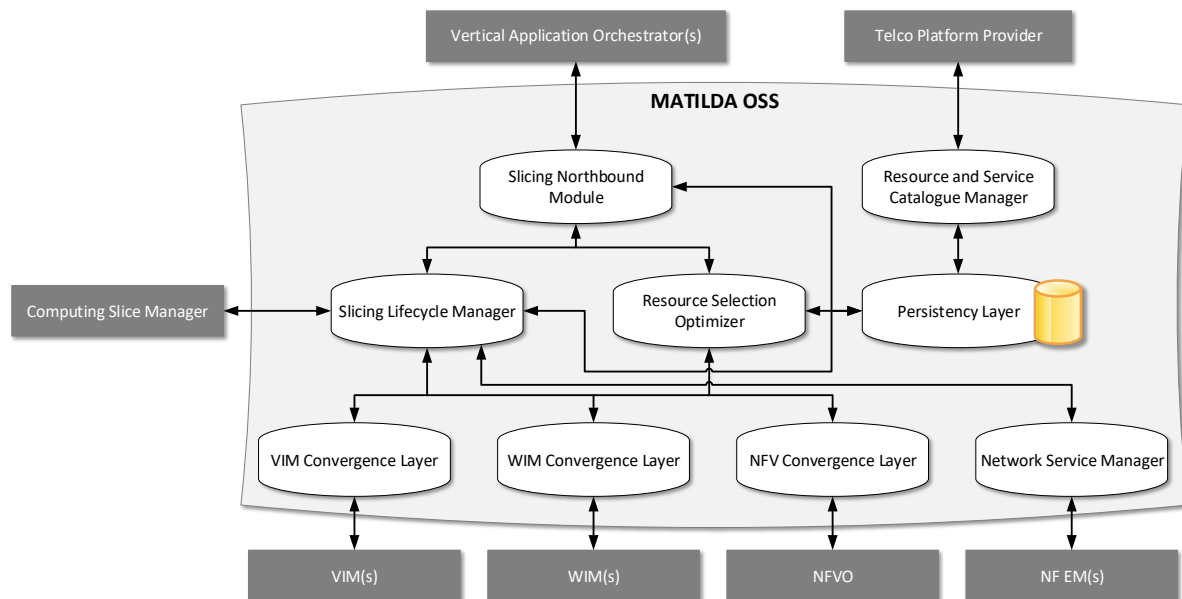


Figure 5: MATILDA OSS/BSS Building Blocks

2.3.3. 5GESSENSE – EmPOWER

5G ESSENCE proposes a highly flexible and scalable 5G small cell network solution by means of leveraging the paradigms of edge cloud computing and Small-Cell-as-a-Service (SCaaS). In such context, it's needed to transform current network implementations to make them much more flexible, nimble, and elastic infrastructures. To that end we have network virtualization as a significant leap to ensure the requirements of the 5G ESSENCE.

With respect to the need of a central entity that manages the interconnections between network functions, the 5G ESSENCE proposes the use of a cloud and Software-Defined (cSD)-RAN controller. The cloud and Software-Defined (cSD)-RAN controller comprises a central entity able to manage the interconnections between network functions. The main advantage of this concept is that various network nodes can be installed on the same infrastructure in such a way that it will be possible to share capacity among them.

Another entity, the Cloud Enabled Small Cell Manager (CESCM) is in charge of interfacing the Edge Cloud infrastructure with the MEC and core services by exposing to the cSD-RAN controller the technical requirements and services needed to instantiate a network slice.

In the scope of 5GESSENCE, the cSD-RAN controller is realized by the 5G-EmPOWER controller. 5G-EmPOWER is an open Mobile Network Operating System for SDN/NFV research and experimentation in heterogeneous mobile networks. Its flexible architecture and the high-level programming APIs allow for fast prototyping of novel services and applications. It is considered as a Multi-access Edge Computing Operating System supporting lightweight virtualization and heterogeneous radio access technologies and it converges SDN and NFV into a single operating

system for multi-access edge computing. 5G-EmPOWER supports multi-tenancy (WiFi and LTE), slicing (WiFi and LTE), and mobility management (WiFi and LTE).

The 5G-EmPOWER provides a northbound interface using REST and Python based API, as shown in Figure 6. With the 5G-EmPOWER Software Development Kit (SDK), different RAN management and network control applications can be developed to control and manage heterogeneous and dense mobile RANs. For the southbound interface, 5G-EmPOWER supports different protocols, including netconf and OpenFlow. Using the southbound interface, the 5G-EmPOWER controller communicates with distributed control agents deployed alongside the CECs and CEAPs. The 5G-EmPOWER control agents exercise control over the radio resource management in the CECs and CEAPs under the directives of the 5G-EmPOWER centralized controller. 5G-EmPOWER natively supports multiple virtual networks (i.e. tenants) sharing the same underlying hardware infrastructure. More specifically, 5G-EmPOWER can slice the radio resources across CECs and CEAPs and tenants can use either the web-based dashboard or REST interfaces to perform CRUD (Create, Retrieve, Update, Delete) operations on network slices.

The slice owners can then use either the web-based management dashboard exposed by the 5G-EmPOWER or can directly interface with the 5G-EmPOWER REST APIs.

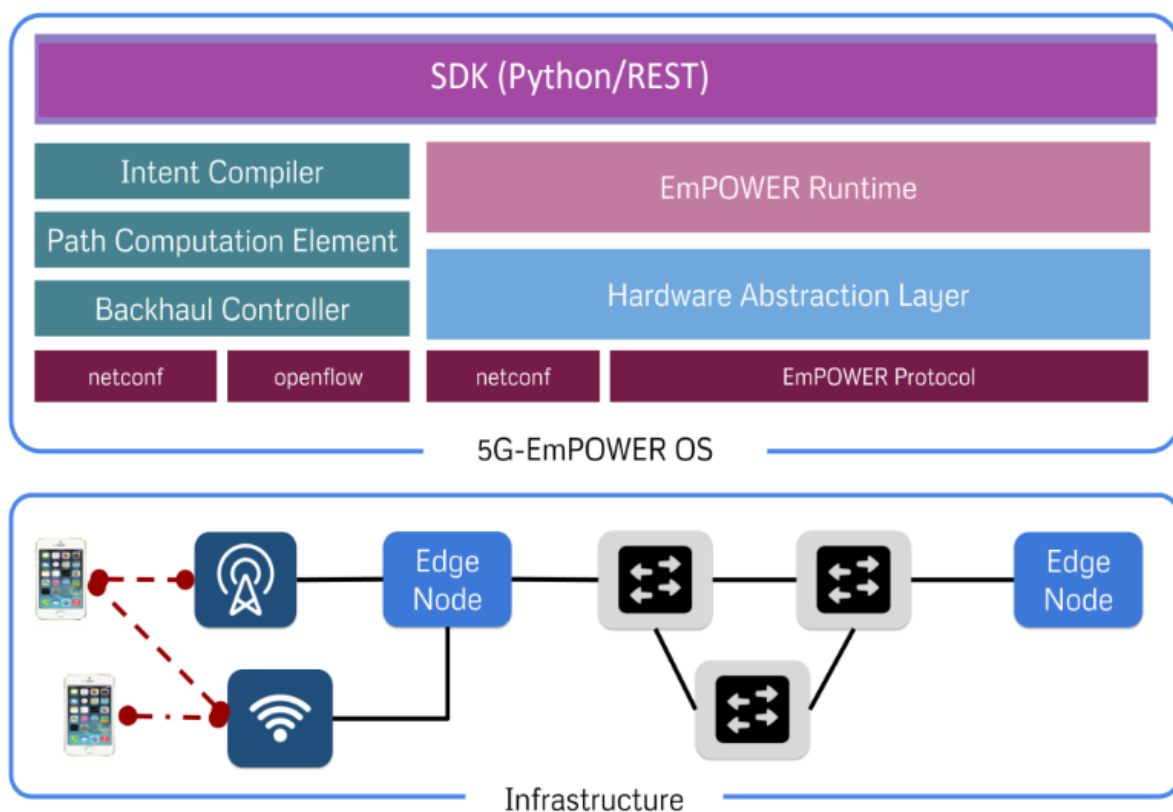


Figure 6: 5G EmPOWER Architecture

3. SLICE MANAGER SPECIFICATIONS

3.1. Relation with overall 5GENESIS requirements

Table 3 summarises the requirements of the Management and Orchestration layer that are related to the Slice Manager functionalities, as they were defined in [1].

Table 3: Requirements of the M&O layer related to the Slice Manager

| | |
|--------------------|--|
| 1 | Resource Catalogue per Service |
| Priority | Essential |
| Description | The M&O layer shall expose an interface to list all resources allocated to a specific VNF service, including the datacentre and edge location of each service and the related infrastructure resources used |
| 2 | Adaptation of Services Scale Up and Down |
| Priority | Essential |
| Description | The M&O layer shall provide interfaces to adapt existing services, so the control application can scale up and down based on monitoring information for optimisation of the resources |
| 3 | Flexible and Fast Allocation of Network Resources |
| Priority | Essential |
| Description | The M&O layer shall be flexible and fast in providing requested resources and the whole end-to-end process must be performed in 90 minutes or less |
| 4 | Distributed NFVI on User or Service Demand |
| Priority | Essential |
| Description | The M&O layer shall support multiple NFVI geographically distributed as PoP (Point of Presence) and must be able to instantiate several VNFs per demand - for example to deploy services as close as possible to the end-users for latency optimisations |
| 5 | Network Service Composition |
| Priority | Essential |
| Description | The M&O layer shall be able to deploy a network service that is composed by both virtual (i.e. VNFs) and physical (i.e. PNF) components and provide lifecycle control |
| 6 | Resource Catalogue per Service |
| Priority | Essential |
| Description | The M&O layer shall support the definition of templates describing the key network slice baseline parameters and resources, oriented to specific |

| | |
|--------------------|---|
| | vertical use cases. This needs to take into account 3GPP slice definitions and 5G NR configuration capabilities |
| 7 | Slice Management |
| Priority | Essential |
| Description | The M&O layer shall provide management and operation of network slice creation across technological domains (i.e. Computing, Network and Radio) based on the provided and exposed infrastructure elements capabilities. It is anticipated that the end-to-end network slice could extent among multiple Facility platforms |
| 8 | Slice Isolation |
| Priority | Essential |
| Description | The M&O layer shall support resource isolation between resources allocated to concurrent network slices. This requirement depends on the isolation capabilities supported at the infrastructure elements and controllers |
| 9 | Slice Stitching and Inter-platform Slice Coordination |
| Priority | Essential |
| Description | The M&O shall support connectivity configuration and traffic steering for slices located on separate platforms with the purpose of creating one end-to-end slice |
| 10 | Coexistence of Multiple Network Slices and/or Services |
| Priority | Essential |
| Description | The M&O layer shall allow the co-existence of multiple slices running concurrently over the same infrastructure |
| 11 | Network Slice Support for User Equipment (UE) |
| Priority | Essential |
| Description | The M&O layer shall allow the platform operator to configure the information which associates a UE to a network slice and a service to a network slice. Based on the subscription, UE capabilities, the access technology being used by the UE, operator's policies and services provided by the network slice, the M&O layer shall also be able to move a UE from one network slice to another, as well as to remove a UE from a network slice |
| 12 | MEC Management and Organisation |
| Priority | Essential |
| Description | M&O layer shall provide means to orchestrate and manage MEC applications deployment and operation on MEC enabled infrastructures at |

| | |
|--------------------|---|
| | the network edge. The solution will follow the recent specification as laid out by ETSI MEC ISG [11] |
| 13 | Real-time Network Monitoring |
| Priority | Essential |
| Description | The M&O layer shall manage and gather monitoring information across all available platform technological domains (NFV, RAN, WAN) and components (eNB, DRAN etc.) in real-time to facilitate the KPI validation objectives |
| 14 | Quality of Service (QoS) Mapping |
| Priority | Optional |
| Description | The M&O layer may be able to map quality of service levels to specific configurations within the network slice |

3.2. Use Cases

This section describes the use cases that will involve the Slice Manager (SM) in the 5GENESIS scope.

3.2.1. Network Slice Instance Creation

| Use Case Stage | Evolution – Specification |
|--------------------------|--|
| Goal | Satisfy request for allocation of a network slice instance with certain characteristics, by creation of new or using existing network slice instance; the request includes the network slice related requirements. |
| Actors and Roles | ELCM |
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Instance • Network Slice Subnet Instance • Transport Network • RAN |

3.2.2. Network Slice Subnet Instance Creation

| Use Case Stage | Evolution – Specification |
|----------------|---|
| Goal | Create a new network slice subnet instance or use an existing network slice subnet instance to satisfy the network slice subnet related requirements. |

| | |
|--------------------------|---|
| Actors and Roles | Slice Manager |
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Subnet Instance • Network Services • NFVI |

3.2.3. Network Slice Instance Termination

| Use Case Stage | Evolution – Specification |
|--------------------------|---|
| Goal | Terminate an existing network slice instance in case it is no longer needed. |
| Actors and Roles | ELCM |
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Instance • Network Slice Subnet Instance • Transport Network • RAN |

3.2.4. Network Slice Subnet Instance Termination

| Use Case Stage | Evolution – Specification |
|--------------------------|--|
| Goal | Terminate or disassociate an existing NSSI which was used by the NSI or NSSI but is no longer needed. Release the allocated resources. |
| Actors and Roles | Slice Manager |
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Subnet Instance • Network Services • NFVI |

3.2.5. Network Slice Instance Modification

| Use Case Stage | Evolution – Specification |
|-------------------------|---|
| Goal | Satisfy a request for modification of an existing network slice instance. |
| Actors and Roles | ELCM |

| | |
|--------------------------|---|
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Instance • Network Slice Subnet Instance • Transport Network • RAN |
|--------------------------|---|

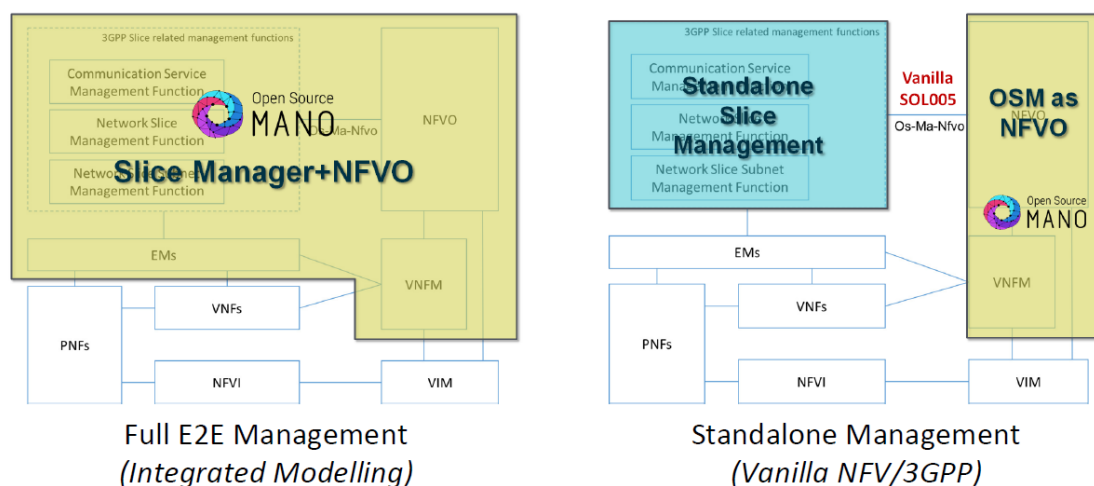
3.2.6. Network Slice Subnet Modification

| Use Case Stage | Evolution – Specification |
|--------------------------|---|
| Goal | Modify an existing network slice subnet instance which is used by the NSI. |
| Actors and Roles | Slice Manager |
| Telecom Resources | <ul style="list-style-type: none"> • Network Slice Subnet Instance • Network Services • NFVI |

4. 5GENESIS SLICE MANAGER

4.1. Overview

Based on the approach presented by ETSI for the Release FIVE of the OSM [5], regarding the implementation of the Slice Manager, there two possible solutions, depicted in Figure 7. The first one requires the extended functionality of the MANO entity, in order to control the full lifecycle of network slices that span across different domains. In the second solution, the Slice Manager is realized by a standalone entity, interacting with the other components of the MANO layer via the defined interfaces.



© ETSI 2017

Figure 7: Slice Manager Solutions by ETSI

5GENESIS has decided to implement the second solution, as it is assumed to fit better to multidomain testbeds, as are the platforms involved in the 5GENESIS, including several core and edge NFV Infrastructures, transport network and physical radio components (PNFs). Figure 8 depicts the Slice Manager element in relation with the other components of the 5GENESIS architecture.

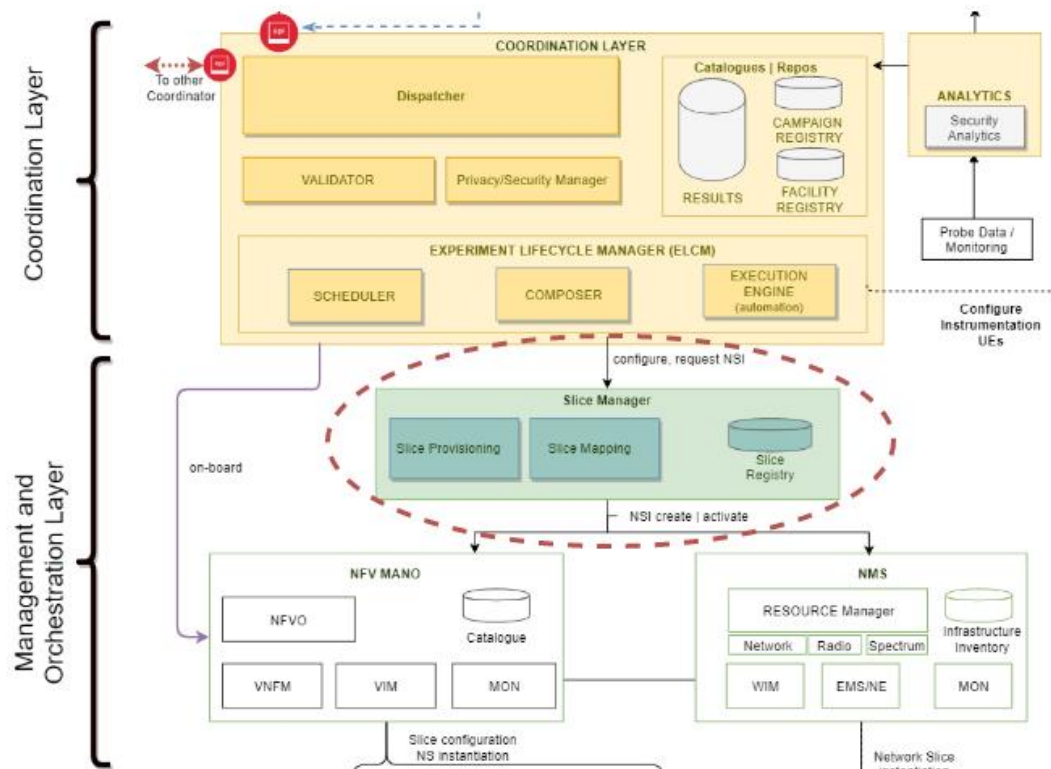


Figure 8: Slice Manager in the 5GENESIS Architecture

4.2. Slice Manager Architecture

The 5GENESIS Slice Manager is based on a highly modular architecture, built as a mess of microservices, each of whom is running on a docker container. The key advantages of this architectural approach are that it offers simplicity in building and maintaining applications, flexibility and scalability, while the containerized approach makes the applications independent of the underlying system. Figure 9 shows an overview of the building blocks that comprise the 5GENESIS Slice Manager.

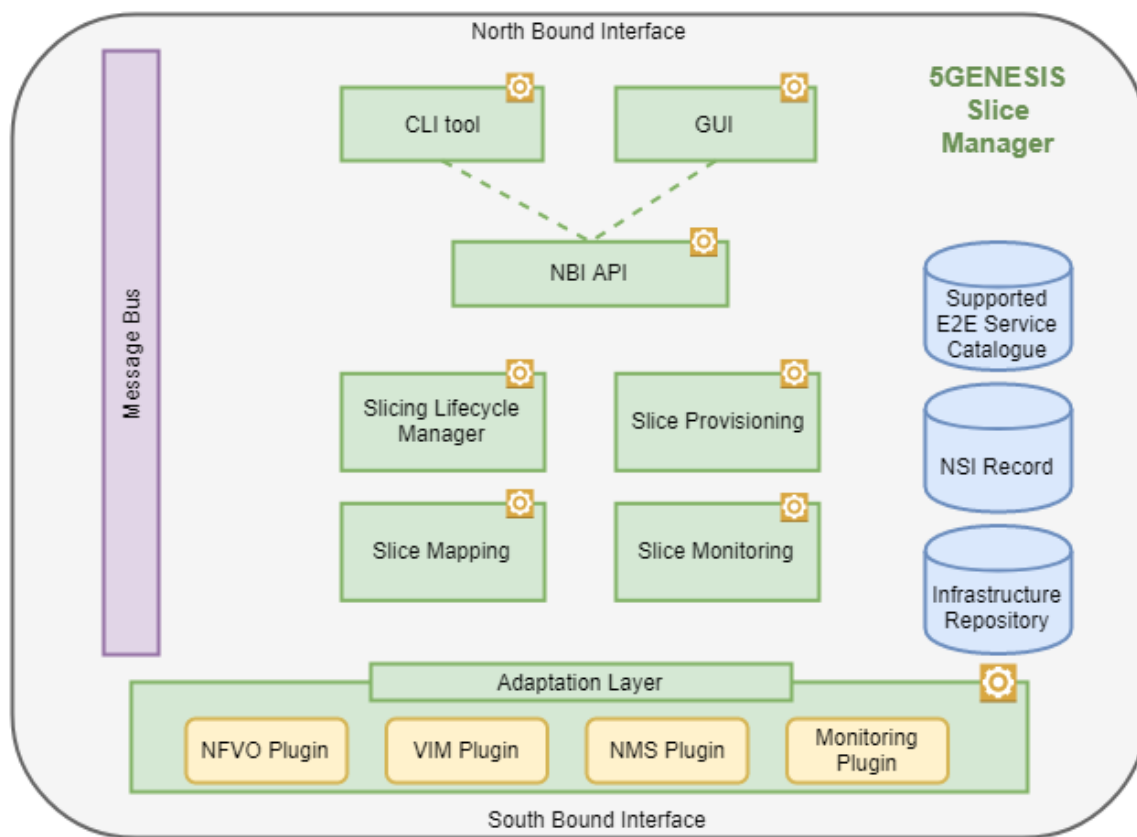


Figure 9: 5GENESIS Slice Manager Building Blogs

4.2.1. Components

4.2.1.1. North Bound Interface API

The North Bound Interface API module implements RESTful APIs, that can be consumed by a component of the coordination layer, e.g. the Experiment Lifecycle Manager, a user/experimenter or the Slice Manager administrator. Refer to Appendix 9.1 for a detailed documentation of the NBI REST APIs. In addition to that, two lightweight modules will be included in Slice Manager packages, allowing a user or an administrator to interact with it, consuming the RESTful APIs.

(a) CLI Tool

The Command Line Tool module creates a group of BASH commands that a user or administrator can run on the terminal of the host where the Slice Manager is installed, in order to interact with it. These commands are built to consume the RESTful APIs that the Slice Manager offers.

(b) GUI

A lightweight web UI will be installed as part of the Slice Manager stack. The web UI will consume the Northbound APIs in order to interact with the Slicing Lifecycle Manager and the repositories. It will also include a guide that will help the Slice Manager user to create a new slice.

4.2.1.2. Slicing Lifecycle Manager

This module is the brain of the 5GENESIS Slice Manager, implementing the Create, Read, Update and Delete (CRUD) functions. The role of this component is twofold. On one hand, it receives requests from the NBI API module and takes any necessary actions for the activation, modification or deactivation of a network slice, as they are described in 4.3. On the other hand, it receives messages from the Slice Monitoring module, regarding the status change of a deployed slice. It interacts with the other components of the Slice Manager, in order to trigger the process that needs to start, depending on the received messages. The 5GENESIS Slicing Lifecycle Manager follows the phases described in [4] and depicted in Figure 10:

- i. Preparation Phase: Includes the creation, validation and the on-boarding of the Network Slice Template (NST), as well as the preparation of the network environment.
- ii. Instantiation, Configuration and Activation phase: Includes the resource provisioning process and the instantiation, configuration and activation of the network functions and components that are part of network slice.
- iii. Run-time Phase: Includes traffic handling, reporting and activities related to modification, such as upgrade, reconfiguration, scaling, etc.
- iv. Decommissioning phase: Includes the deactivation of the NSI and the release of network resources that were used as part of that.

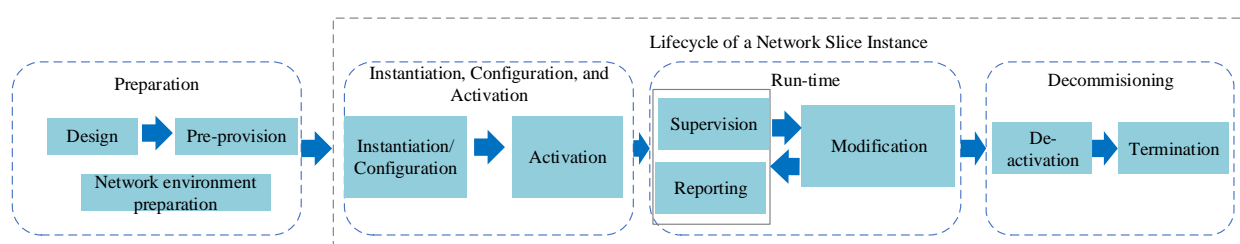


Figure 10: Lifecycle phases of an NSI

4.2.1.3. Slice Mapping

This module hosts a very important process that runs during the slice creation phase, the placement process. This process is responsible for optimally selecting the infrastructure resources to be used for a new slice, based on the slice requirements, as they are described in the NST, and the available resources of the infrastructure layer. The operations of this process are described in detail in the Activity Diagram below.

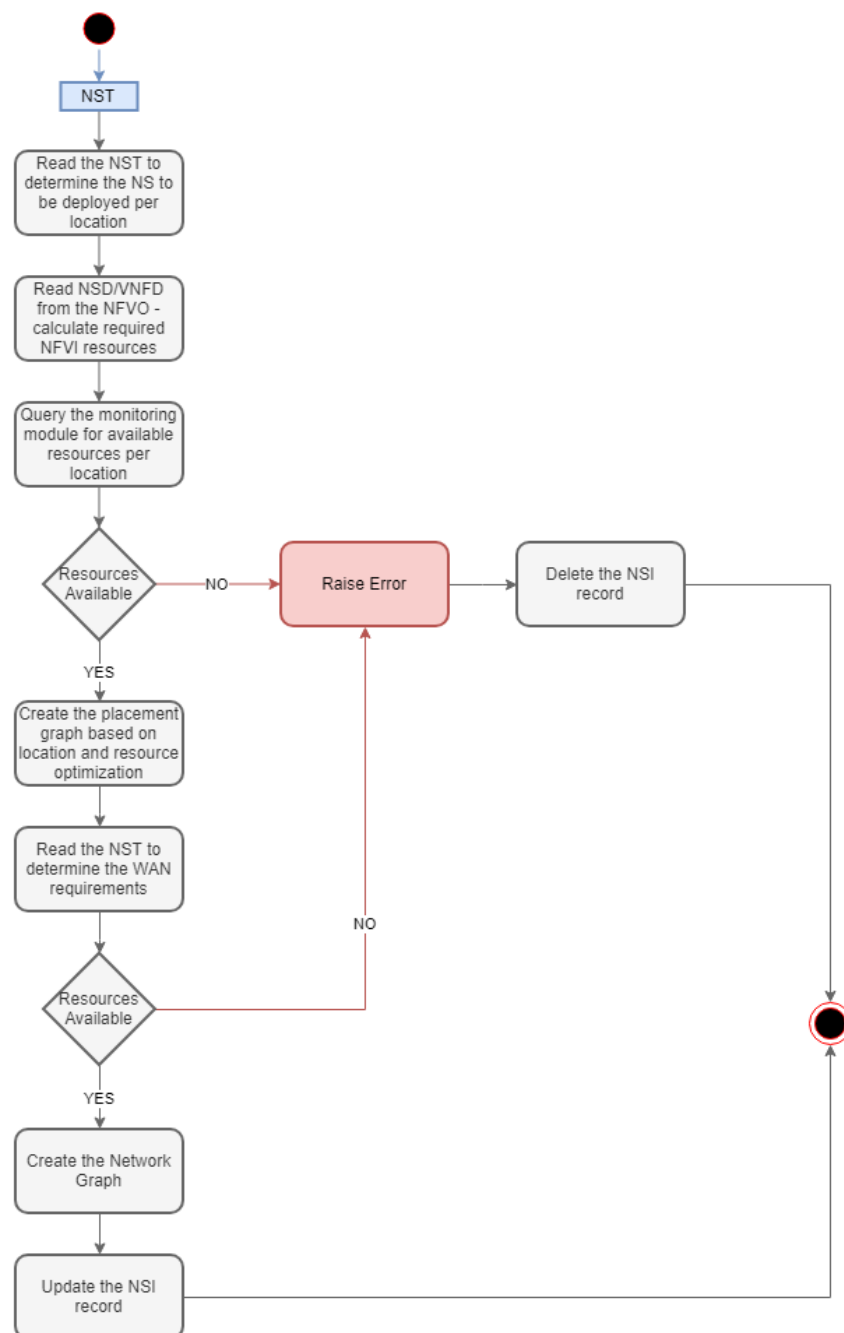


Figure 11: Placement Process Activity Diagram

If the placement process finishes successfully, the Slicing Lifecycle Manager has the responsibility to request the activation of reserved resources and services. To do this, it interacts with Slice Provisioning block.

4.2.1.4. Slice Provisioning

The Slice Provisioning module receives requests from the Slicing Lifecycle Manager service in order to set up, configure or delete the Wide Area Network paths, the isolated NFVI tenant spaces and all the required Network Services, to configure the radio component parameters

and register the newly created slice to the Monitoring system. It does so by using the VIM, NFVO, NMS and Monitoring Plugins of the Adaptation Layer.

4.2.1.5. Slice Monitoring

The Slice Monitoring module is responsible for monitoring the health and the status of every deployed slice. It uses the Adaptation Layer plugins to send status check messages to the MANO components below the Slice Manager and reports any slice status change to the Slicing Lifecycle Manager service.

4.2.1.6. Adaptation Layer

The Adaptation Layer module provides a level of abstraction regarding the underlying layer technology, making it feasible for the Slice Manager to operate over any MANO layer component without any modifications to its core functionality, as long as the proper plugin has been loaded. This module is comprised of VIM, NFVO, NMS and Monitoring plugins, one for each of the MANO layer components that the Slice Manager supports. The responsibility for each plugin is to receive request messages from the Slicing Lifecycle Manager and translate them to the proper API call of the supported component. After that, it is in charge of properly handling the responses from the underlying components to the API calls.

4.2.1.7. Databases

Slice Manager uses three storage repositories, that are detailed described below.

(a) Infrastructure Repository

The Infrastructure Repository is a database that stores details about the MANO layers components which are in use in the platform. More specifically, it stores the necessary information, e.g. URL, username, password, etc. that Slice Manager needs in order to interact with the deployed VIMs, NFVO and NMS on the platform. The database is populated by the Slice Manager administrator before the instantiation of new slices. Slicing Lifecycle Manager interacts with these databases in order to query or add information.

(b) NSI Record Repository

Network Slice Instance (NSI) Record Repository is a database that stores the NSI record for every created Network Slice. Each NSI record is created, updated or removed by the Slicing Lifecycle Manager. When the Slicing Lifecycle Manager receives a request with an NST for the instantiation of a new Network Slice, it creates a new NSI based on the NST, included information about the reserved resources, the placement, the status and the deployment time of the Slice.

(c) Supported E2E Service Catalogue

This database optionally stores predefined End to End Services that the infrastructure layer supports. They contain information regarding the VNFs, the PNFs, the Network Graph, locations, etc. that the platform can support. The purpose of this information is on one hand to simplify the NST of a new slice by simply mapping it to a predefined end-to-end service, on the other hand to help the Slicing Lifecycle Manager to take optimal decisions regarding the placement of Network Services, the transport network graph, etc. It is populated by the Slice Manager administrator.

4.2.1.8. Message Queueing software

Message queuing software, often called message broker or queue manager, is a software that acts as the middleman for the various services of the Slice Manager, creating queues with the messages that these services exchange with each other. The advantage of this approach is that the exchange of messages will be significantly faster when the Slice Manager will have to deal with multiple requests at a short period of time.

4.3. Slice Lifecycle Operations

This section describes in detail the Create, Read, Update and Delete (CRUD) functions of Network Slices, that the 5GENESIS Slice Manager supports. Table 4 presents the endpoints and methods defined in order to trigger these functions. The base URL prefix for each endpoint is `http://<slice_manager_ip>:8000`.

Table 4: Network Slice Functions Endpoints

| Function | Method | URL | Data |
|----------------------|--------|-----------------------|------|
| Create Network Slice | POST | /api/slice | NST |
| Read Network Slice | GET | /api/slice/<slice_id> | - |
| Update Network Slice | PUT | /api/slice/<slice_id> | NST |
| Delete Network Slice | DELETE | /api/slice/<slice_id> | - |

4.3.1. Slice Creation

Table 5 describes the steps that comprise the workflow during the Network Slice creation phase, while Figure 12 depicts the Sequence Diagram.

Table 5: Slice Creation Steps

| Step | Description |
|------|---|
| 1 | Slice Lifecycle Manager (SLM) receives a request for a new Network Slice creation and the NST. |
| 2 | SLM responds with a generated unique id of this Slice. |
| 3 | SLM creates a new NSI record which stores it in the NSI Record Repository. |
| 4 | SLM queries the Supported E2E Service Catalogue using the Network Slice Service type from the NST. If it finds a match, it uses the Service information for the Placement process in step 5. |
| 5 | SLM queries the Monitoring system for the available resources on the Infrastructure Layer. |
| 6 | SLM updates the NSI status to "Placement". |
| 7 | SLM starts the placement process, based on the required resources, available resources and the registered services that match the NST service (if any), creating the placement graph and the transport network graph. |
| 8 | SLM updates the NSI status to "Resource Provisioning". |
| 9 | SLM interacts with the VIMs in the placement graph via the suitable VIM plugin of the Abstraction Layer in order to create a new isolated tenant for the slice, with the appropriate quotas. It gets back the credentials for each newly created tenant and updates the NSR. |
| 10 | SLM interacts with the WIM via the suitable WIM plugin of the Abstraction Layer, sending the transport network graph in order for the WIM to reserve the WAN resources and enable the connectivity between end points using SDN rules. |
| 11 | SLM interacts with the Element Management System (EMS) via the suitable plugin of the Abstraction Layer in order to set the parameters for the radio elements that will be part of the Network Slice (either PNFs or VNFs). |
| 12 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to register the new Tenants. |
| 13 | SLM updates the NSI status to "Instantiation". |
| 14 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to instantiate the Network Services on the new Tenants, based on the placement graph. Waits until the NSs are in running state and then queries the NFVO for the Network Service Record (NSR) to get the management and data IPs of each VNF. |
| 15 | SLM interacts with the Monitoring system in order to register the new Network Services, using the management IPs from step 14. |
| 16 | SLM updates the NSI status to "Activation". |
| 17 | SLM interacts with the WIM in order to activate the traffic steering between the end points that are part of the slice. |

| | |
|----|---|
| 18 | SLM interacts with the EMS in order to activate the radio component services. |
| 19 | SLM updates the NSI status to “Running”. |

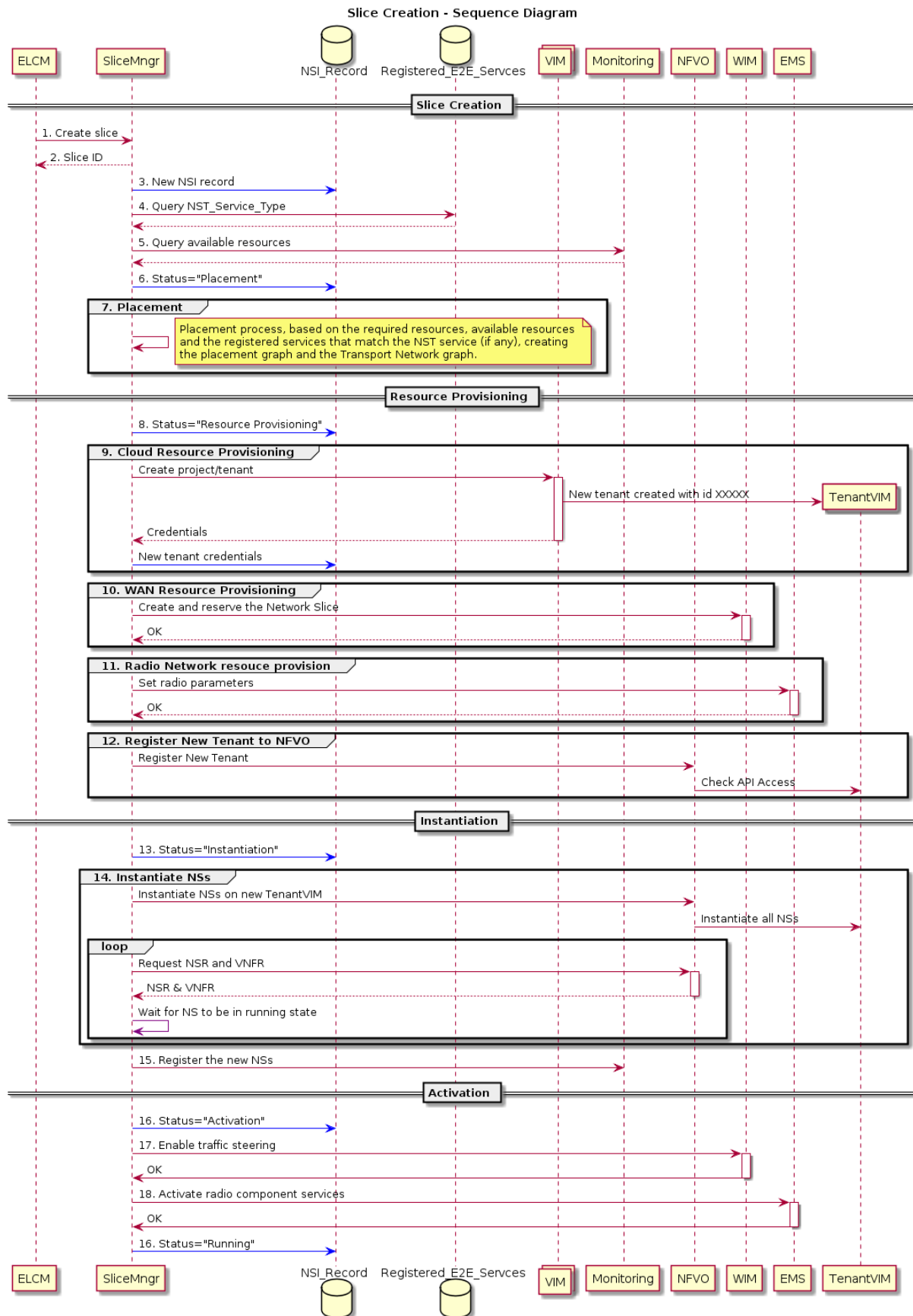


Figure 12: Slice Creation Sequence Diagram

4.3.2. Slice Read

Table 6 describes the steps that comprise the workflow during the Network Slice Record read phase, while Figure 13 depicts the Sequence Diagram.

Table 6: Slice Read Steps

| Step | Description |
|------|--|
| 1 | SLM receives a Read request with a Slice ID. |
| 2 | SLM queries the NSI Record Repository for an NSI that matches the requested Slice ID. |
| 3 | If Step 2 returns a result, SLM responds with re requested NSI Record. Else it responds with an error message. |

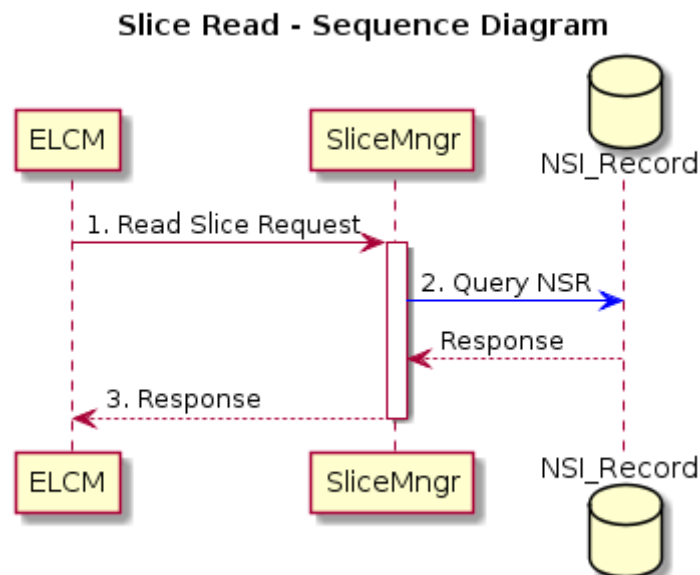


Figure 13: Slice Read Sequence Diagram

4.3.3. Slice Update

A Slice Update is requested by sending a new NST to the Slice Manager containing information of the Slice that is to be updated. Slice Manager compares the new NST with the existing NSI record of the running slice and proceeds with the necessary modifications. These modifications might be:

- Instantiate new Network Services on existing or new Tenants of the Slice
- Update or delete running Network Services
- Modify the transport network parameters
- Start/Stop/Update services on existing radio components of the Slice

Table 7 describes the steps that comprise the workflow during the Slice update phase, while Figure 14 depicts the Sequence Diagram.

Table 7: Slice Update Steps

| Step | Description |
|---|---|
| 1 | SLM receives an Update request with a Slice ID and an NST containing information about the updated slice. |
| 2 | SLM queries the NSI Record Repository for an NSI that matches the requested Slice ID. |
| 3 | If Step 2 returns a result, SLM responds that the Slice is updating. Else it responds with an error message. |
| 4 | If Step 2 is successful, SLM updates the NSI status to “Updating”. |
| 5 | SLM queries the Monitoring system for the available resources on the Infrastructure Layer. |
| A. New NSs need to be instantiated | |
| A.1 | SLM starts the placement process for the new NSs, based on the available resources, the resource requirements and the Registered end-to-end services (if any) and extracts the new placement graphs and the new transport network graph (if necessary). |
| A.2 | If new VIMs are to be added to the network slice after the placement graph is created, SLM interacts with them via the suitable VIM plugin of the Abstraction Layer in order to create a new isolated tenant for the slice, with the appropriate quotas. It gets back the credentials for each newly created tenant and updates the NSR. It then registers the new tenants to the NFVO. |
| A.3 | SLM interacts with VIMs that are already part of the Slice in order to modify the quotas (if necessary). |
| A.3 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to instantiate the Network Services on the new Tenants, based on the placement graph. Waits until the NSs are in running state and then queries the NFVO for the Network Service Record (NSR) to get the management and data IPs of each VNF. |
| A.4 | SLM interacts with the Monitoring system in order to register the new Network Services. |
| A.5 | SLM interacts with the WIM in order to add and activate flows to and from the new end points of the slice (if necessary). |
| A.6 | SLM interacts with the EMS in order to configure and activate the new radio services (if necessary). |
| B. Update/Delete running NSs | |

| | |
|---|---|
| B.1 | SLM interacts with VIMs that are already part of the Slice in order to modify the quotas (if necessary). |
| B.2 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to customize/delete the running NSs |
| C. Update transport network parameters | |
| C.1 | SLM interacts with the WIM in order to add/remove or customize flows in order to meet the requirements. |
| D. Update/Start/Stop radio services | |
| D.1 | SLM interacts with the EMS in order to update/start/stop a radio service. |
| 6. | SLM updates the NSI status to “Running” and stores the new NSI Record. |

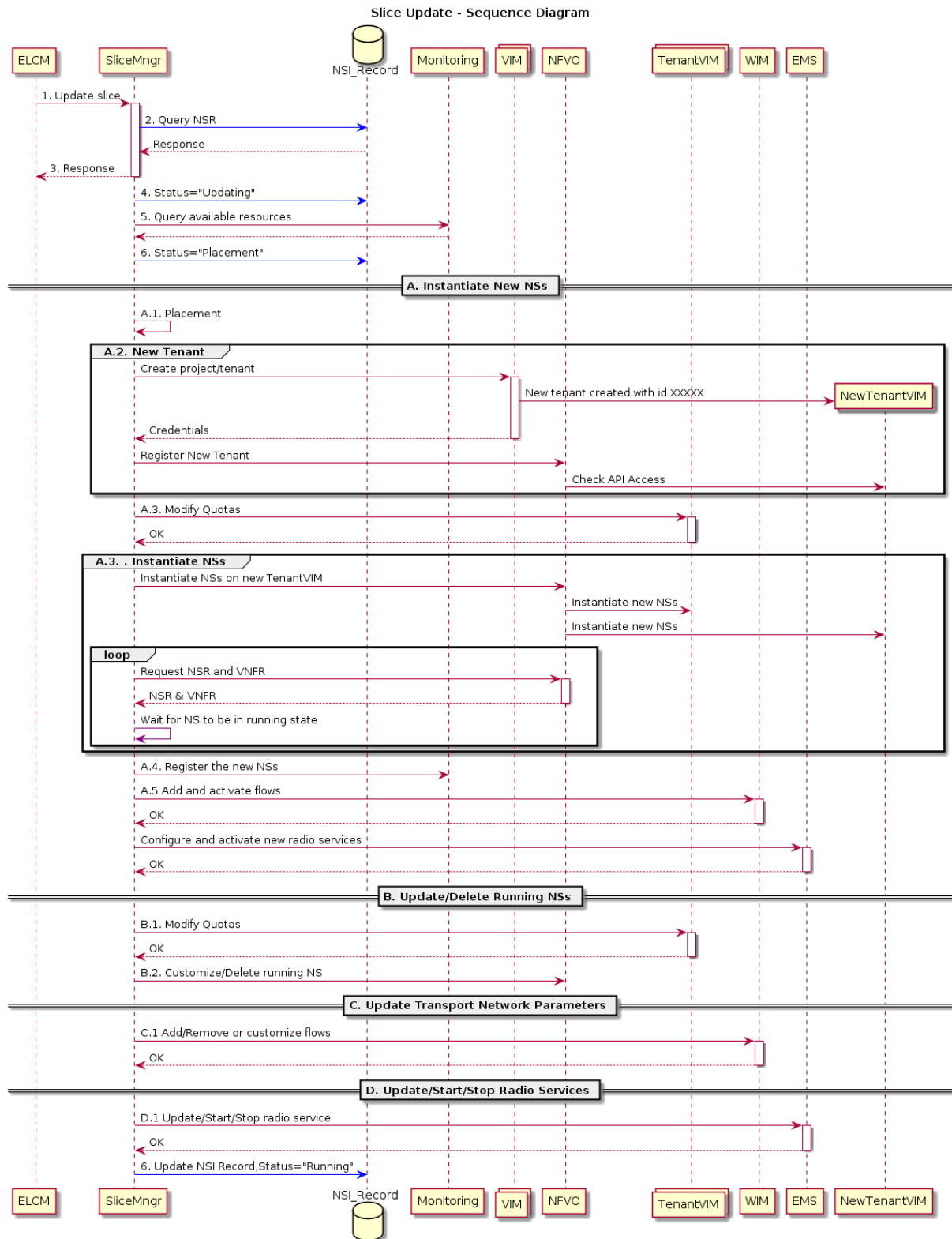


Figure 14: Slice Update Sequence Diagram

4.3.4. Slice Deletion

Table 8 describes the steps that comprise the workflow during the Slice deletion phase, while Figure 15 depicts the Sequence Diagram.

Table 8: Slice Deletion Steps

| Step | Description |
|------|---|
| 1 | SLM receives a Delete request with a Slice ID. |
| 2 | SLM queries the NSI Record Repository for an NSI that matches the requested Slice ID. |
| 3 | If Step 2 returns a result, SLM responds that the Slice is being removed. Else it responds with an error message. |
| 4 | If Step 2 is successful, SLM updates the NSI status to “Deleting”. |
| 5 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to delete the Network Services running on this Slice. Waits until the NSs are deleted. |
| 6 | SLM interacts with the NFVO via the suitable plugin of the Abstraction Layer in order to delete the registered Tenant VIMs. |
| 7 | SLM interacts with the VIMs via the suitable plugin of the Abstraction Layer in order to delete the Tenants of this Slice and any other related element, such as subnets, routers, security groups etc. |
| 8 | SLM interacts with the WIM via the suitable plugin of the Abstraction Layer in order to release the reserved transport network resources for this slice. |
| 9 | SLM interacts with the EMS via the suitable plugin of the Abstraction Layer in order to stop all the radio related services that were running. |
| 10 | SLM removes the NSI Record of the slice from the NSI Record Repository. |

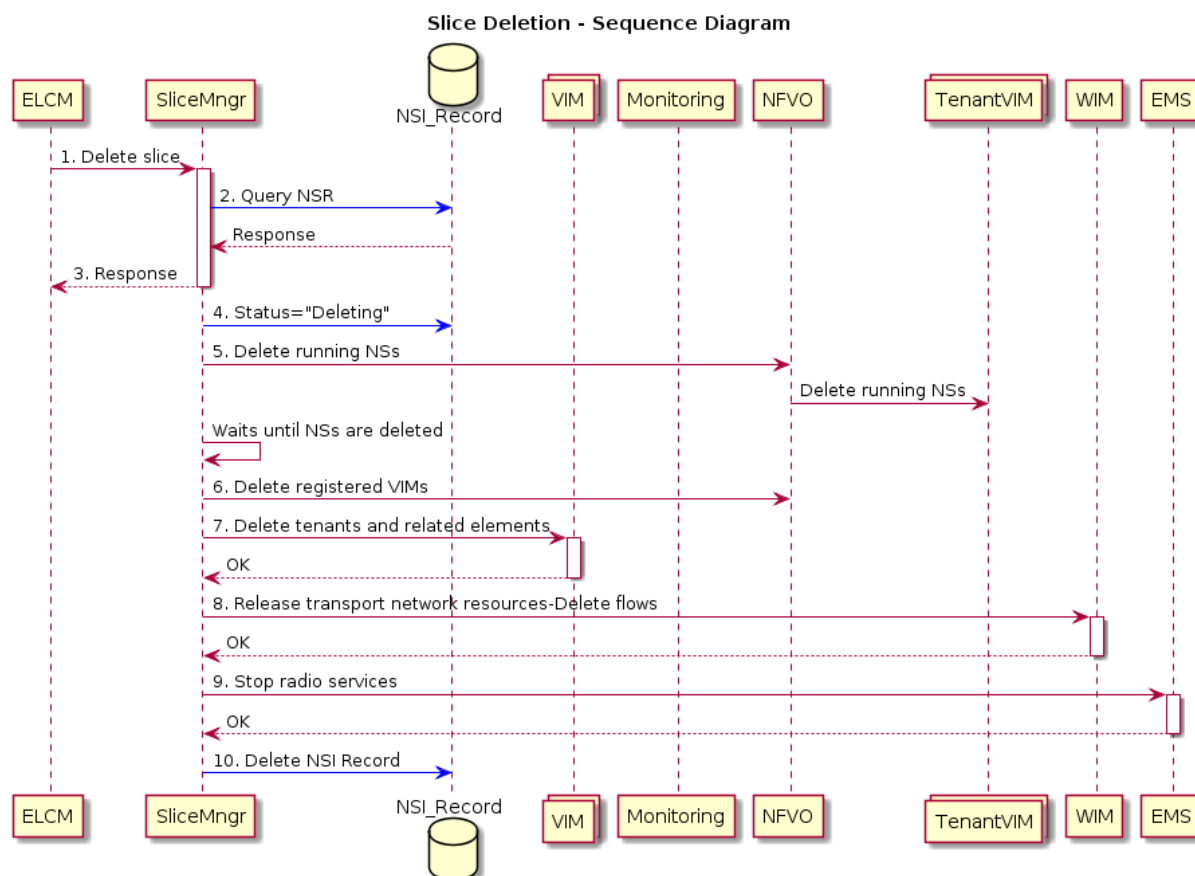


Figure 15: Slice Deletion Sequence Diagram

4.4. Registered End-to-End Services

Registered End-to-End services are used to provide information regarding the underlying infrastructure and the services that it is able to host. The information is described in configuration files that are onboarded to the Slice Manager by the platform administrator.

The registered end-to-end services serve two purposes. Firstly, Slice Manager uses the onboarded configuration files during the instantiation of an NSI, in order to take decisions regarding the placement of NSs, the transport network graph and QoS. Secondly, they allow NST files, which describe the actual NSIs that will be created, to be less complex, since some configuration parameters are already on-boarded to the Slice Manager.

Note that the Registered end-to-end services are optional. If Slice Manager receives an NST that contains a service type which is not matched with any of the registered services, it will take the aforementioned decisions based on the NST parameters and the default choices.

5. INTERFACES

5.1. Slice Manager Interfaces

Figure 16 represents the interfaces that are defined in order for the Slice Manager to interact with the other components of the 5GENESIS architecture. The interfaces implement some of the functionalities of the reference points that are described by the 3GPP in Specification 28.500 - Telecommunication management; Management concept, architecture and requirements for mobile networks that include virtualized network functions [12].

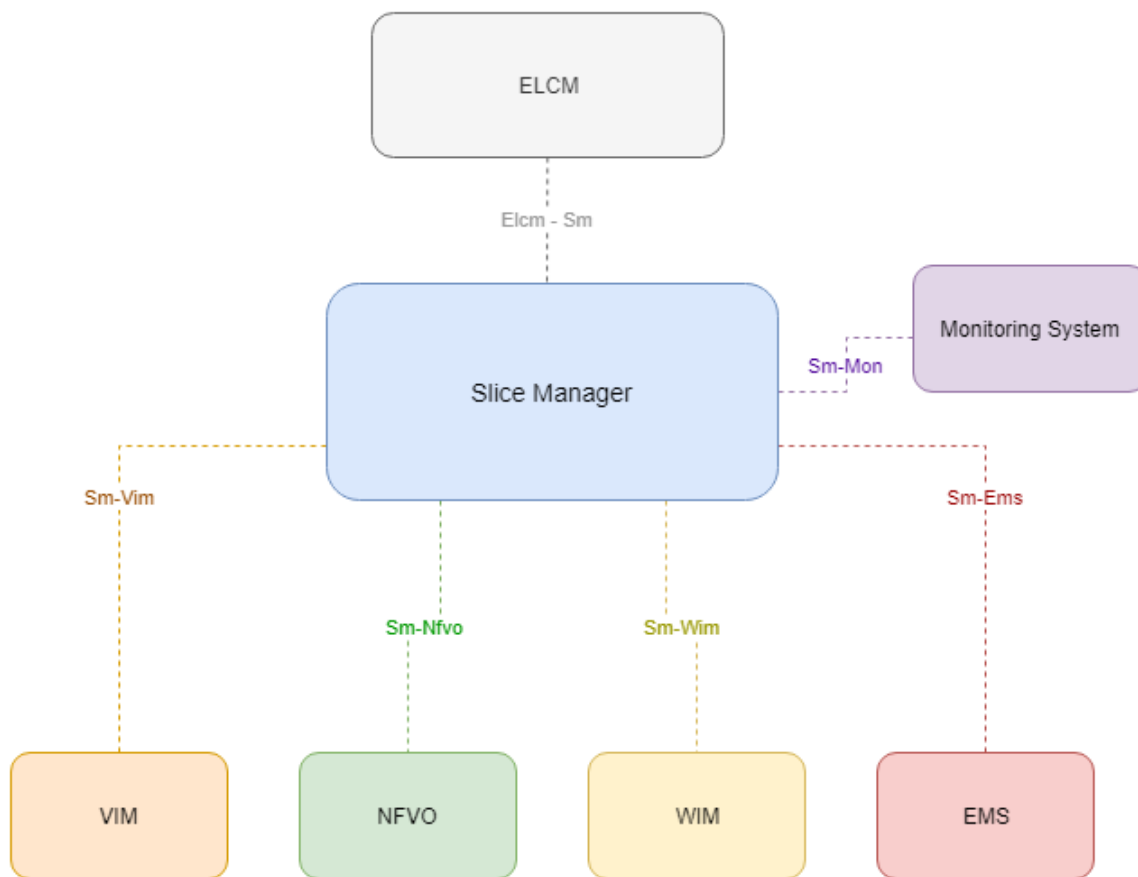


Figure 16: Slice Manager Interfaces

5.1.1. Elcm-Sm

This interface implements the interconnection between the ELCM of the coordination layer and the Slice Manager. The role of the ELCM is the execution of the experiments and is described in detail in D3.15 [13]. Through this interface, the ELCM sends requests to the NBI of the Slice Manager regarding the Lifecycle Management of the network slices.

From the technological aspect, the interconnection between the components is achieved with the use of Test Automation Platform (TAP) plugins, which realizes the execution engine of the ELCM. Currently, the supported plugins support the following actions:

- ssh action: The ELCM uses the ssh action plugin in order to connect to the Slice Manager and use the CLI tool in order to execute slice CRUD functions.
- scp action: This action is used by the ELCM in order to pass the NST to the Slice Manager during the slice creation/modification phase.

5.1.2. Sm-Vim

This interface implements the interconnection between the Slice Manager and the VIM. The Slice Manager uses this interface in order to create, modify or delete the isolated NFVI tenant spaces for each slice. The implementation is based on the type of the VIM and is achieved with the use of the appropriate plugin on the Adaptation layer of the Slice Manager, as described in 4.2.1.6.

5.1.3. Sm-Nfvo

This interface implements the interconnection between the Slice Manager and the NFVO. It realizes some of the functionalities of the Os-Ma-nfvo reference point described in [12]. The implementation depends on the type of the NFVO and is achieved with the use of the appropriate plugin. Slice Manager uses this interface to interact with the NFVO in order to:

- Read the onboarded VNF Descriptors (VNFD) and NS Descriptors (NSD) to determine the requirements for every new slice.
- Register or delete NFVI tenant for a slice.
- Create, configure or delete NSs.
- Read the NS Record (NSR) and the VNF Record (VNFR) for every deployed NS.

5.1.4. Sm-Wim

This interface implements the interconnection between the Slice Manager and the WIM, using the WIM plugin of the adaptation layer. Through this interface, the Slice Manager sends to WIM:

- Request resource provisioning along with the transport network requirements for a new slice, including the network graph and QoS parameters, such as latency, bandwidth, etc.
- Requests for enabling or disabling the traffic steering.

5.1.5. Sm-Ems

This interface implements the interconnection between the Slice Manager and the EMS, using the WIM plugin of the adaptation layer. This interface is part of the Itf-N reference point described in [12]. The Slice Manager interacts with the EMS through this interface in order to:

- Configure parameters of the radio services that the EMS will manage on the radio domain components.
- Activate, modify or deactivate radio services.

5.1.6. Sm-Mon

This interface implements the interconnection between the Slice Manager and the Monitoring System of the infrastructure. Through this interface, the Slice Manager sends requests to the Monitoring System in order to query the available resources of the available NFVI per domain and register/unregister new VNF instances.

5.2. Integration with the other 5GENESIS Components

Slice Manager integration with the other MANO layer components of the 5GENESIS architecture is achieved with the use of the plugins in the adaptation layer module, as described in 4.2.1.6. In addition to that, the Slice Manager creates NBI endpoints through which the platform administrator can register, modify or unregister components of the platform's MANO layer. You can refer to Appendix 9.1 for a detailed description of these endpoints.

The Platform Administrator can reach these endpoints using REST API calls to the Slice Manager instance, the CLI tool or the GUI that are installed along with the Slice Manager.

For the registration of new MANO layer component to the Slice Manager, a file containing information of this component, e.g. type, URL, username, password, etc., has to be sent in addition to the request. Appendix 9.2 presents the template file for each component.

6. RELEASE A OF 5GENESIS SLICE MANAGER

6.1. Supported features and functionalities

This section describes the feature and functionalities of the Slice Manager that are implemented with Release A.

6.1.1. Supported features and functionalities

Table 9 presents the current status in Release A of the 5GENESIS Requirements regarding the Slice Manager, as they were described in Table 3.

Table 9: Implemented 5GENESIS Requirements in Release A

| 1 | Resource Catalogue per Service |
|----------------------------|--|
| Release A Implementation | Preliminary |
| Implementation Description | In Release A, Slice Manager stores the information regarding the resources per VNF and PNF in the NSI inventory. An interface for the retrieving the information must be defined and implemented. |
| 2 | Adaptation of Services Scale Up and Down |
| Release A Implementation | Work in Progress |
| Implementation Description | Release A of the Slice Manager does not support scaling capabilities for resource optimization |
| 3 | Flexible and Fast Allocation of Network Resources |
| Release A Implementation | Implemented |
| Implementation Description | Release A of the Slice Manager is able to deploy slices based on the components that were integrated in the platforms during phase 1 of the project in the target time |
| 4 | Distributed NFVI on User or Service Demand |
| Release A Implementation | Implemented |
| Implementation Description | Slice Manager is able to deploy a multi-vim and multi-location slices, e.g. for edge service deployment, using the underlying WIM for the configuration of the transport network between the different locations |
| 5 | Network Service Composition |

| | |
|-----------------------------------|--|
| Release A Implementation | Implemented |
| Implementation Description | In Release A, Slice Manager is able to deploy and apply day-1 and day-2 configuration in both VNFs and PNFs. The PoC described in 6.3.1 involves the deployment of a vEPC VNF and an eNodeB PNF |
| 6 | Resource Catalogue per Service |
| Release A Implementation | Preliminary |
| Implementation Description | Slice Manager follows the definitions of network slicing defined by 3GPP in [4]. Moreover, the NST scheme for describing network slicing parameters is well defined by the Slice Manager NBIs, although more fields referring to 5G NR might have to be added in the next phases |
| 7 | Slice Management |
| Release A Implementation | Implemented |
| Implementation Description | The creation of end-to-end slices above multiple facility platforms is demonstrated in the PoC described in 6.3.1 |
| 8 | Slice Isolation |
| Release A Implementation | Preliminary |
| Implementation Description | The complete isolation between slices is achieved on the NFVI domains, where the slice manager creates a new tenant for every new slice to be deployed. Nevertheless, isolation is still to be implemented in the transport network domain and in the RAN domain |
| 9 | Slice Stitching and Inter-platform Slice Coordination |
| Release A Implementation | Work in Progress |
| Implementation Description | Release A of the Slice Manager does not support coordination between different slices |
| 10 | Coexistence of Multiple Network Slices and/or Services |
| Release A Implementation | Preliminary |
| Implementation Description | Release A of the Slice Manager supports the creation of concurrent slices over the same infrastructure, as long as this is available by the underlying components |
| 11 | Network Slice Support for User Equipment (UE) |
| Release A Implementation | Work in Progress |

| | |
|-----------------------------------|--|
| Implementation Description | Release A does not support User Equipment functionalities |
| 12 | MEC Management and Organisation |
| Release A Implementation | Work in Progress |
| Implementation Description | Orchestration on MEC enabled infrastructure at the network edge is currently under development |
| 13 | Real-time Network Monitoring |
| Release A Implementation | Work in Progress |
| Implementation Description | Slice Manager in Release A is not offering per slice monitoring capabilities across multiple domains |
| 14 | Quality of Service (QoS) Mapping |
| Release A Implementation | Work in Progress |
| Implementation Description | The option for defining QoS parameters for a new slice is under development |

The Release A of a the 5Genesis Slice Manager includes most of the architectural components presented in Section 4.2. More Specifically:

- All three databases, i.e. NSI record repository, Infrastructure repository and the supported End-to-End Services Catalogue, are included in Release A. Mongo DB [14] is adopted as the technological solution for the implementation of the repositories, taking advantage of the Document Oriented Storage feature.
- The NBI API module that creates the RESTful APIs is implemented with the use of Python and the Flask framework¹. Moreover, a swagger tool that visualizes the API Specification definitions in an interactive UI has been created for documentation purposes. Figure 17 depicts the UI of the swagger tool. It can be reached via a web browser using the URL <http://<slice_manager_IP> :8001.
- Two lightweight modules implementing the Slice Manager CLI tool and web UI (depicted in Figure 18) have been included in Release A. Python, Flask framework and the Click² library are the technologies that have been used for the implementation of both tools. Note that while the CLI tools implements all of the required functionalities, the UI is in a preliminary version, implementing only part of these functionalities.

¹ <https://palletsprojects.com/p/flask/>

² <https://click.palletsprojects.com/en/7.x/>

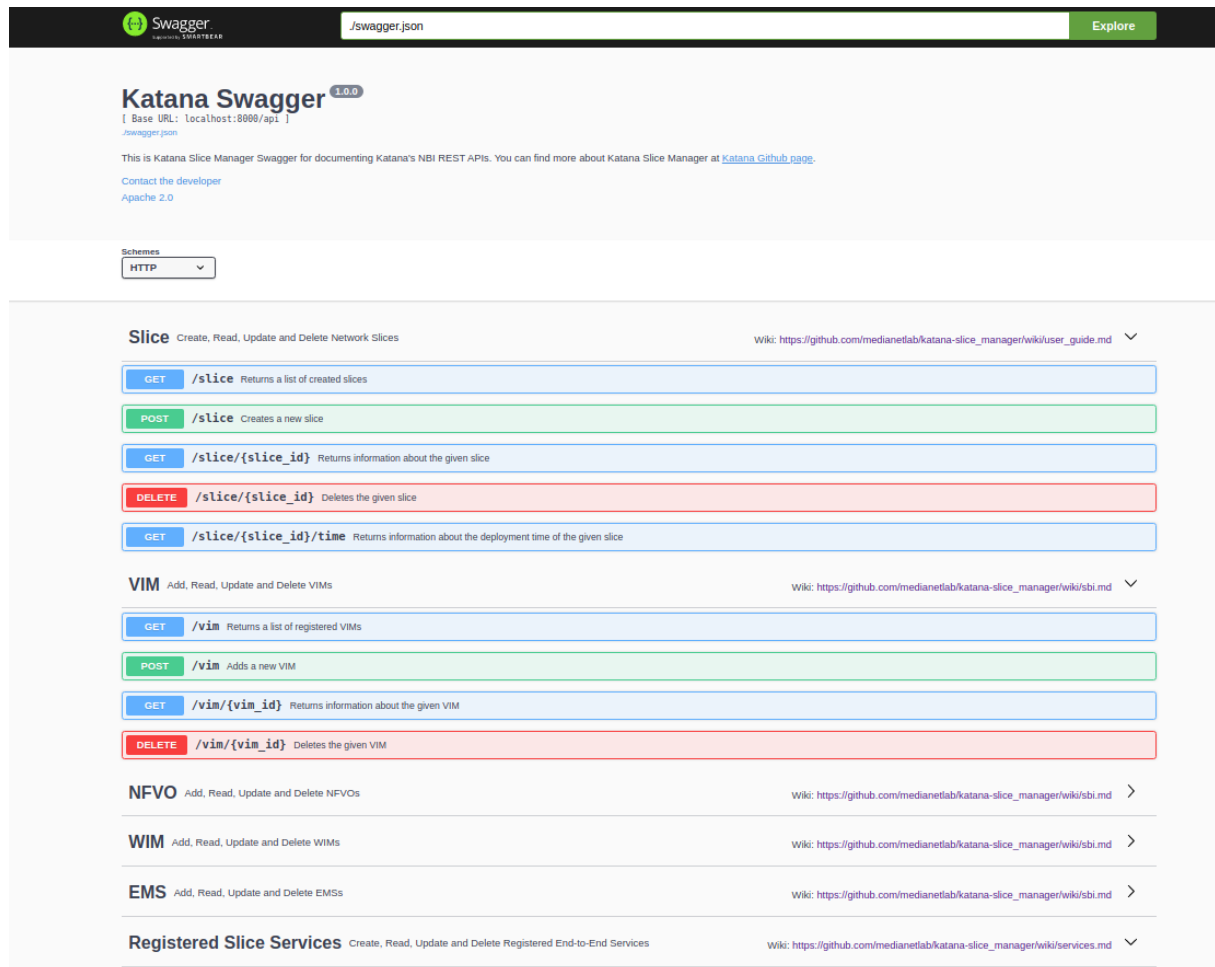


Figure 17: Slice Manager Swagger tool

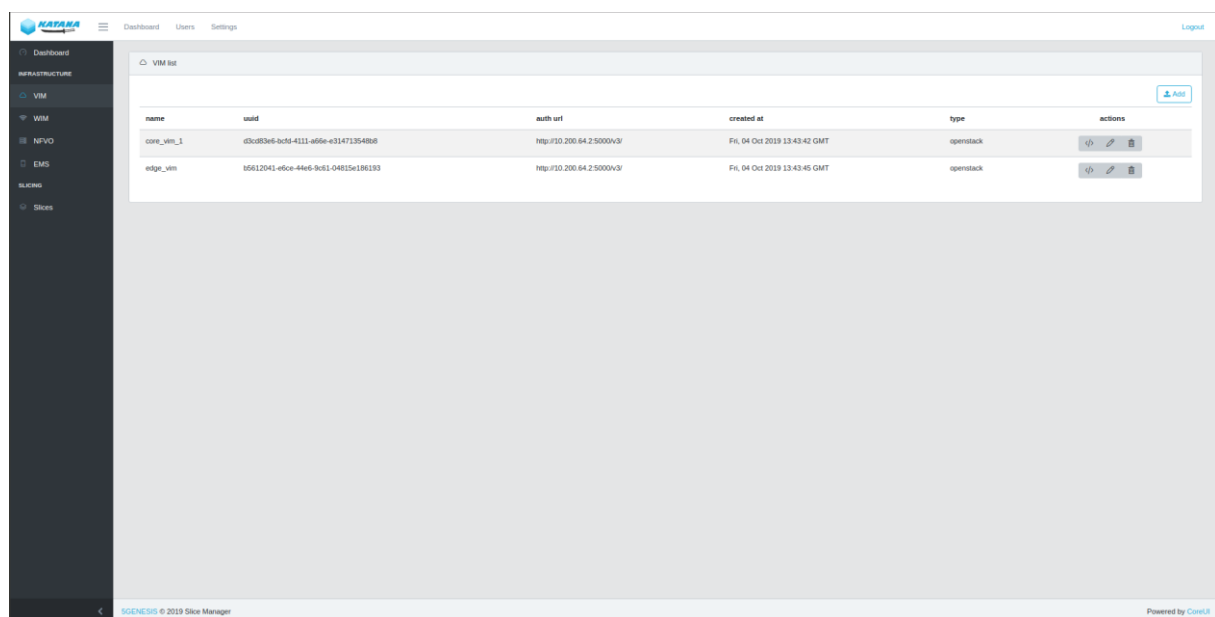


Figure 18: Slice Manager UI

- Regarding the core components of the Slice Manager, only the Slice Monitoring module is not included in Release A and will be implemented in an updated version. Slice Lifecycle Manager, Slice Provisioning and Slice Mapping are implemented as dockerized components in the Slice Manager stack. Flask framework and several Python libraries were used also for these modules.
- Several plugins of the abstraction layer have been packaged in the Slice Manager repository. These plugins are used for the communication with the South Bound components that are supported in Release A and described in Section 6.1.2.

6.1.2. Supported MANO layer components

Table 10 presents the MANO layer components that have been integrated with the current release of the Slice Manager. The target is to add the necessary plugins so that the SM will be able to interact with all the MANO layer component of the 5GENESIS platforms.

Table 10: Supported MANO Components

| MANO Component | Type | Platform |
|----------------|----------------------------|----------|
| NFVO | OSM Release FIVE | Athens |
| | | Malaga |
| VIM | Openstack | Athens |
| | | Malaga |
| | OpenNebula | Malaga |
| WIM | Custom Athens Platform WIM | Athens |
| EMS | Custom Athens Platform EMS | Athens |

6.1.3. Network Slice Template

The Network Slice Template (NST) is used for describing the parameters of a new network slice. Appendix 9.4 describes the scheme of the NST that has been defined and used by the Slice Manager in Release A. Note that the scheme might change in future implementations when new components of the 5G NR will be integrated in the platform infrastructures.

6.2. Integration and deployment

6.2.1. Deployment and integration at Athens platform

Figure 19 depicts the deployment of the Slice Manager in Athens Platform and the integration with the other components of the 5GENESIS testbed. In this deployment, Slice Manager has been integrated and tested with OSM Release FIVE, OpenStack VIM, simple implementations of the WIM and EMS developed for the Athens Platform [15].

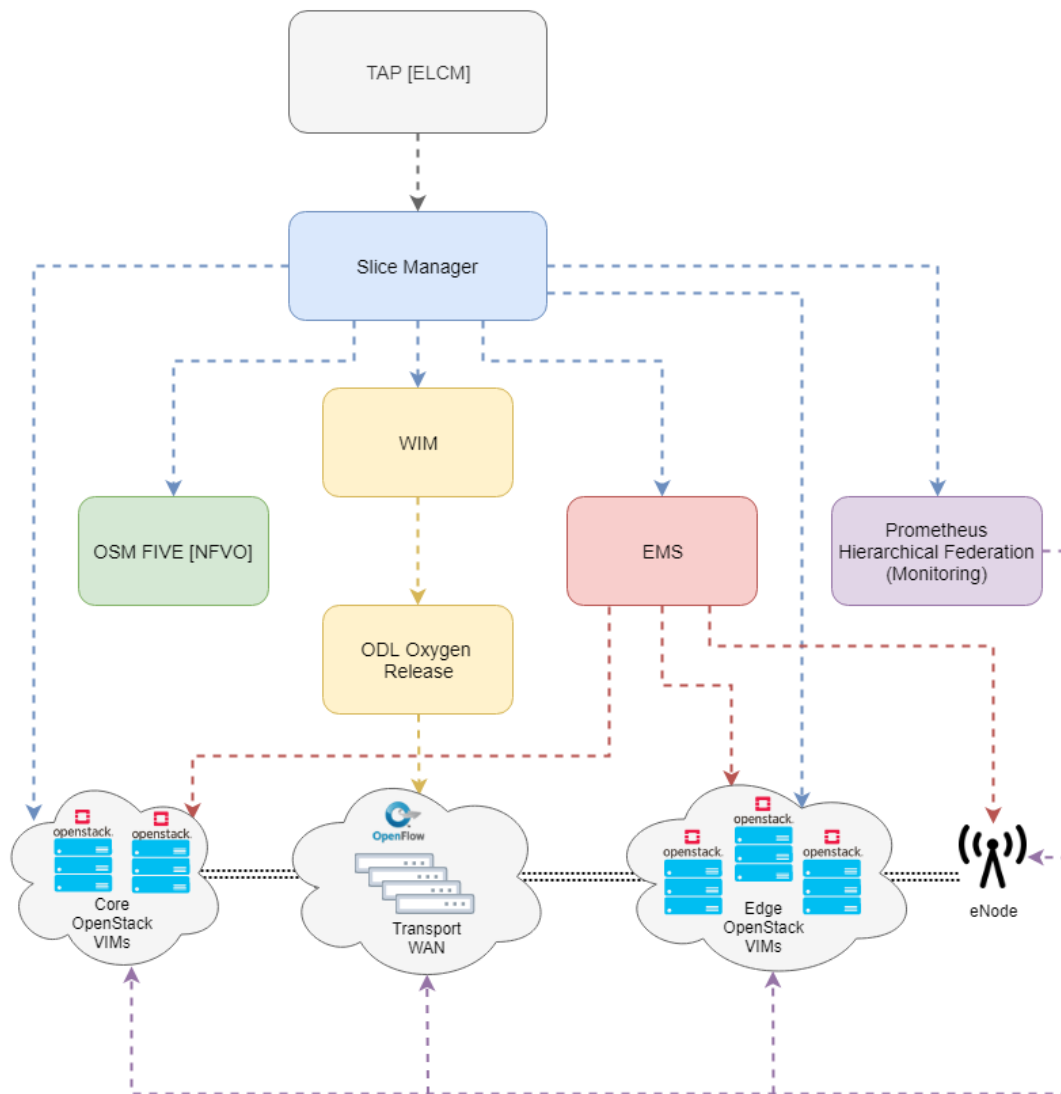


Figure 19: Slice Manager Deployment in Athens Platform

On the technical part, the Slice Manager has been installed on a Virtual Machine (VM), alongside the other VMs where the rest MANO layer components are installed. The hardware configuration of the VM is:

- RAM: 8 GBytes
- VCPUs: 4
- Storage: 40 GBytes
- Network Interfaces: 1

6.2.2. Deployment and integration at Malaga platform

The Slice Manager has been integrated in the Malaga Platform within the Phase 1 of the programmed schedule. Figure 20 depicts the current Malaga Platform deployment, including the integration with the other components of the 5GENESIS testbed. In this deployment, Slice Manager has been integrated and tested with OSM Release FIVE, OpenStack core VIM, Open Nebula edge VIM and a beta release of the ELCM.

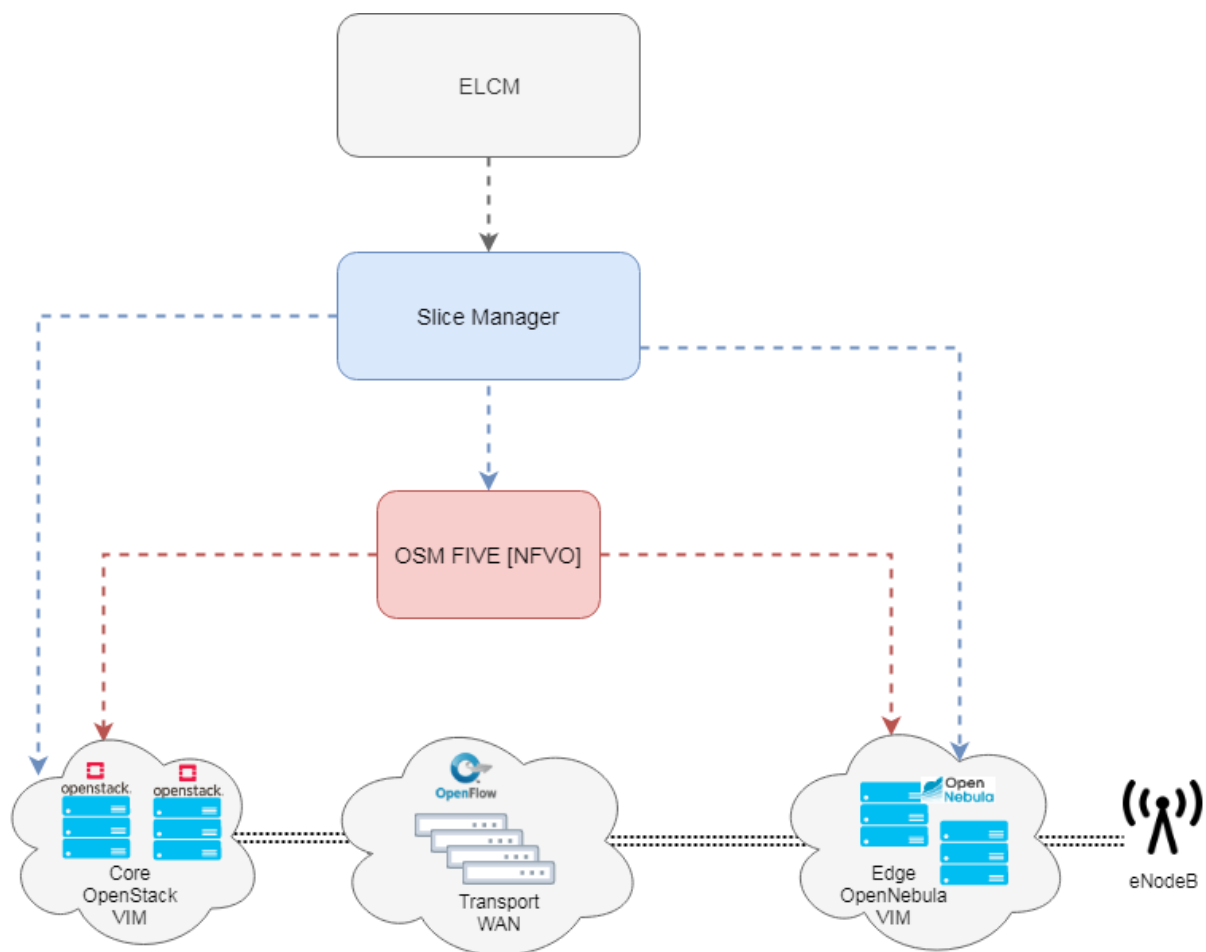


Figure 20: Slice Manager Deployment in Malaga Platform

On the technical part, in a similar way as the Athens Platform, the Slice Manager has been installed on a Virtual Machine (VM), alongside the other VMs where the rest MANO layer components are installed. The hardware configuration of the VM is:

- RAM: 4 GBytes
- VCPUs: 2
- Storage: 40 GBytes
- Network Interfaces: 1

6.3. Preliminary Testing

6.3.1. Proof of Concept

A proof-of-concept demonstration was held on the NCSRD testbed that involved the creation of an end-to-end 4G Service with the use of Slice Manager. The service comprised of an

Amarisoft virtual EPC (vEPC) NS and a physical Amarisoft eNodeB located on an edge location of the NCSRD testbed. Table 11 presents the steps of the slice creation.

Table 11: Slice Creation Steps

| Steps | Description |
|-------|---|
| 1 | Platform administrator registers the MANO layer components that will be used by the SM, i.e. OSM NFVO, Openstack as both core and edge VIMs, custom Athens Platform WIM and custom Athens platform EMS. |
| 2 | The end-to-end 4G service is registered to the SM, indicating the supported NS per location and the network graph. |
| 3 | TAP, the execution engine of the ELCM, sends a slice creation request, along with the NST that describes the slice requirements and the service that it will support. |
| 4 | SM calculates the required resources for the NSs that are defined in the NST. |
| 5 | SM starts the placement process and defines the NFVI where the vEPC NS will be deployed. |
| 6 | SM interacts with the core Openstack and creates a new tenant. |
| 7 | SM interacts with the OSM and registers the new tenant VIM. |
| 8 | SM interacts with the WIM in order to reserve the resources and activate the path between the vEPC and the eNodeB. |
| 9 | SM requests the OSM to initiate the vEPC NS on the new tenant VIM. |
| 10 | SM requests the WIM to activate the traffic steering between the eNodeB and the vEPC |
| 11 | SM interacts with the EMS in order to configure the RAN parameters that will be used by the vEPC and the eNodeB |

6.3.2. Slice Creation Time

Slice Manager is used to measure the NSI creation time. In addition to that, it also measures the time needed for the:

- Placement process.
- Resource provisioning.
- Deployment of all the NSSIs that are part of the NSI.
- Configuration of the transport network via the WIM.
- Configuration and activation of the RAN services via the EMS.

The information regarding the slice creation time can be reached via a REST API at the endpoint `http://<slice_manager_IP>:8000/api/slice/<slice_id>/time`. You can refer to Appendix 9.1.1 for further details of the API. Figure 21 depicts an example of a NSI deployment time on the Athens Platform.

```
themis@themis-ncsrd:~/Documents/5genesis/Config-files$ katana slice deployment_time a8102353-9dfe-4616-850a-1ba2ac7fd71f
{
  "Slice_Deployment_Time": "45.5129",
  "Placement_Time": "0.0053",
  "Provisioning_Time": "5.1888",
  "NS_Deployment_Time": {
    "vepc": "30.1980"
  },
  "WAN_Deployment_Time": "0.1316",
  "Radio_Configuration_Time": "10.0351"
}
```

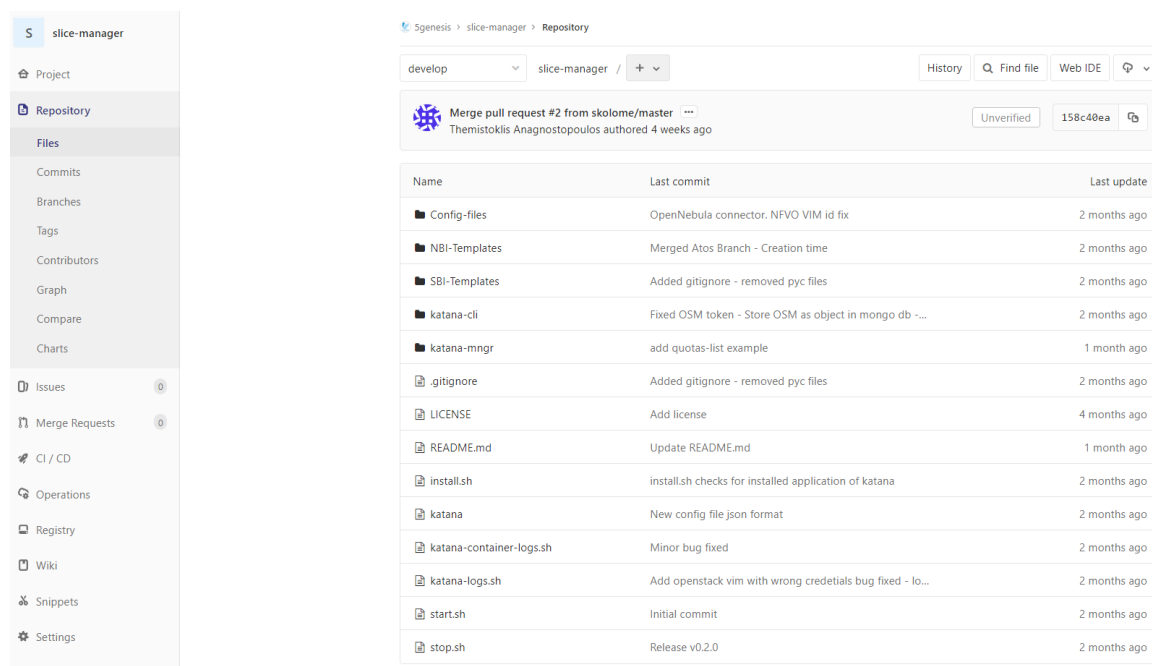
Figure 21: Slice Deployment Time

7. RELEASE A SUMMARY AND FUTURE PLANS

This document presented the activities related to the realization of network slicing in the 5GENESIS framework. It briefly presented network slicing concepts and solutions, before describing specifications, functional use cases, interfaces and architectural design of the 5GENESIS Slice Manager.

The activities of Task 3.2 kicked off on Month 7, and a preliminary version of the Slice Manager was available by Month 11. Two more minor Releases followed, implementing some more feature and fixing some bugs, before the first stable Release 1.0.0 was published, after it was integrated and tested at Athens and Malaga platforms. The software is available to all the partners of the 5GENESIS consortium at the project's Gitlab repository (Figure 22). Moreover, a documentation Wiki (Figure 23) has been created, including detailed information about the Slice Manager and instructions on how to deploy and use it. Release A features are described in detail in Section 6.

The name chosen for the implementation of the 5GENESIS Slice Manager is “Katana”. It is published under the Apache License 2.0. Each release follows the Semantic Version 2.0.0, described in detail in Appendix 9.3.



| Name | Last commit | Last update |
|--------------------------|--|--------------|
| Config-files | OpenNebula connector, NFVO VIM id fix | 2 months ago |
| NBI-Templates | Merged Atos Branch - Creation time | 2 months ago |
| SBI-Templates | Added gitignore - removed pyc files | 2 months ago |
| katana-cli | Fixed OSM token - Store OSM as object in mongo db -... | 2 months ago |
| katana-mngr | add quotas-list example | 1 month ago |
| .gitignore | Added gitignore - removed pyc files | 2 months ago |
| LICENSE | Add license | 4 months ago |
| README.md | Update README.md | 1 month ago |
| install.sh | install.sh checks for installed application of katana | 2 months ago |
| katana | New config file json format | 2 months ago |
| katana-container-logs.sh | Minor bug fixed | 2 months ago |
| katana-logs.sh | Add openstack vim with wrong credentials bug fixed - lo... | 2 months ago |
| start.sh | Initial commit | 2 months ago |
| stop.sh | Release v0.2.0 | 2 months ago |

Figure 22: Slice Manager Gitlab Page

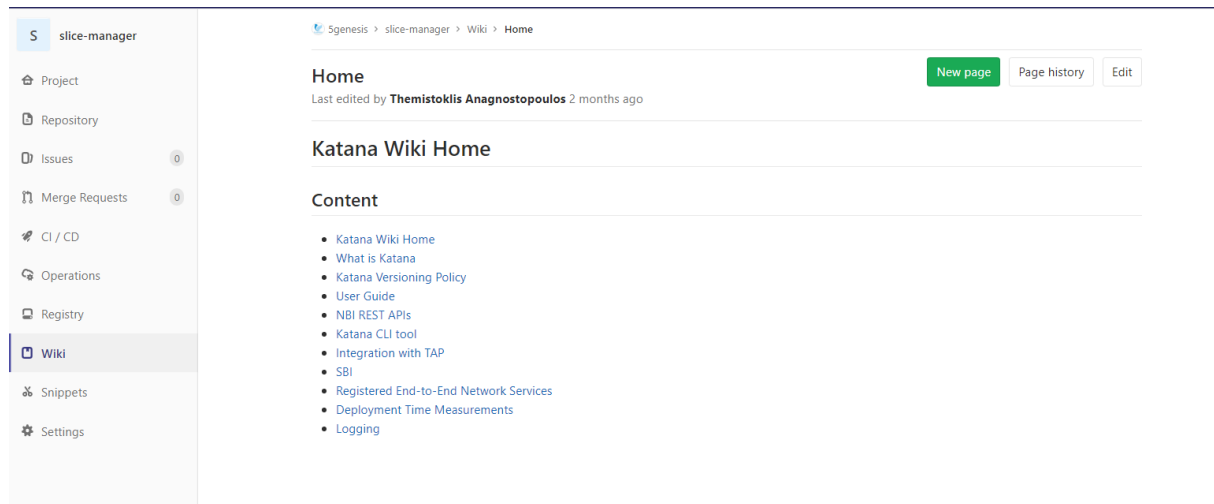


Figure 23: Slice Manager Wiki Page

Over the next minor and major Releases, the rest of the required features and functionalities will be implemented, while many of the existing ones will significantly improve. Moreover, Slice Manager will evolve to support all different versions and implementations of the underlying components of the 5GENESIS architecture. This will enable the deployment and integration of the Slice Manager to every platform. The next version of this deliverable, D3.4, will present the final release of the Slice Manager and the integration to each platform.

8. REFERENCES

- [1] 5GENESIS Consortitum, "D2.1 Requirements of the Facility," 2018. [Online]. Available: https://5genesis.eu/wp-content/uploads/2018/11/5GENESIS_D2.1_v1.0.pdf.
- [2] 5GENESIS Consortium, "D2.2 Initial overall facility design and specifications," 2018. [Online]. Available: https://5genesis.eu/wp-content/uploads/2018/12/5GENESIS_D2.2_v1.0.pdf.
- [3] 5GENESIS Consortium, "D2.3 Initial planning of tests and experimentation," [Online]. Available: https://5genesis.eu/wp-content/uploads/2018/12/5GENESIS_D2.2_v1.0.pdf.
- [4] 3GPP, "28.801 - Telecommunication management; Study on management and orchestration of network slicing for next generation network," 2018. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3091>.
- [5] ETSI - OSM, "Introduction to NFV and OSM," 2017. [Online]. Available: <http://osm-download.etsi.org/ftp/osm-5.0-five/5th-hackfest/presentations/5th%20OSM%20Hackfest%20-%20Session%200%20-%20Introduction%20to%20NFV%20and%20OSM.pdf>.
- [6] Linux Foundation, "ONAP," [Online]. Available: <https://wiki.onap.org>.
- [7] 5GTANGO, "5GTANGO," [Online]. Available: <https://wiki.onap.org>.
- [8] Sonata, "TANGO Slice Manager," [Online]. Available: <https://github.com/sonata-nfv/tng-slice-mngr/wiki>.
- [9] MATILDA, "MATILDA," [Online]. Available: <https://www.matilda-5g.eu/>.
- [10] MATILDA Consortium, "Network and Computing Slice," September 2019. [Online]. Available: <https://private.matilda-5g.eu/documents/PublicDownload/487>.
- [11] "ETSI MEC Specifications," [Online]. Available: <https://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing>.
- [12] 3GPP, "Specification 28.500 - Telecommunication management; Management concept, architecture and requirements for mobile networks that include virtualized network functions," June 2018. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2935>.
- [13] 5GENESIS, "D3.15 - Expirement Life Cycle Manager," September 2019. [Online].
- [14] MongoDB, [Online]. Available: <https://www.mongodb.com/>.
- [15] 5GENESIS Consortium, "D3.1 - Management and Orchestration," September 2019. [Online].

9. APPENDICES

9.1. North Bound Interface REST APIs

You can reach Katana Slice Manager at `http://<katana_ip>:8000` using the REST APIs listed below.

9.1.1. Network Slice

9.1.1.1. Endpoints

| Method | URL | Data | Description |
|--------|----------------------------|------|--|
| GET | /api/slice | - | Returns a list of slices and their details |
| GET | /api/slice/<slice_id> | - | Returns the details of specific slice |
| POST | /api/slice | NST | Adds a new network slice – Responses with the Slice ID |
| GET | /api/slice/<slice_id>/time | - | Returns Deployment Time report |

9.1.1.2. Examples

```
echo $(curl -X POST -H "Content-Type: application/json" -d
@slice_example.json http://10.100.129.104:8000/api/slice)

b8a8465d-3499-458c-bfda-5610e5dab8f1
```

```
curl -XGET http://10.100.129.104:8000/api/slice | python -m json.tool
```

```
[
  {
    "_id": "b8a8465d-3499-458c-bfda-5610e5dab8f1",
    "created_at": 1559819045.742592,
    "nsi": {
      "description": "test-slice",
      "id": "test-slice",
      "name": "test-slice",
      "nsd-ref": [
        {
          "id": "a9ac7f6e-9b44-4f99-860b-db4c94794b4d",
          "name": "vepc",
          "short-name": "vepc"
        }
      ],
      "radio-ref": {
        "location": "minilab"
      },
      "short-name": "test-slice",
      "type": "embb",
    }
  }
]
```

```

        "wim-ref": {
            "bidirectional": true,
            "endpoint-segment": [
                {
                    "ip": "10.201.0.101",
                    "service_id": "enodeb",
                    "service_name": "enodeb"
                },
                {
                    "service_id": "internet",
                    "service_name": "enodeb"
                }
            ],
            "link_params": {
                "bw": "N",
                "latency": "N"
            },
            "topology": "ELAN"
        },
        "status": "Activation"
    }
]

```

9.1.2. NFVO

9.1.2.1. Endpoints

| Method | URL | Data | Description |
|--------|---------------------|------------------|---|
| GET | /api/nfvo | - | Returns a list of NFVO and their details |
| GET | /api/nfvo/<nfvo_id> | - | Returns the details of specific NFVO |
| POST | /api/nfvo | NFVO Config file | Adds a new NFVO instance - Responses with the NFVO ID |
| DELETE | /api/nfvo/<nfvo_id> | - | Removes an NFVO instance |
| PUT | /api/nfvo/<nfvo_id> | NFVO Config file | Updates a registered NFVO info |

9.1.2.2. Examples

```

echo $(curl -X POST -H "Content-Type: application/json" -d @nfvo.json
http://10.100.129.104:8000/api/nfvo)
98e1b60e-7d2f-491e-9341-3b464243c408

```

```

url -XGET http://10.100.129.104:8000/api/nfvo | python -m json.tool
{

```

```

    "_id": "98e1b60e-7d2f-491e-9341-3b464243c408",
    "config": {
      "network": "flat",
      "network-mode": "provider",
      "wim-sap": "wim-name-1"
    },
    "created_at": 1559818152.5329616,
    "description": "This is a real NFVO to be added",
    "location": "rack-5, NCSRD",
    "name": "nfvo-real-osm",
    "nfvoip": "10.200.64.55",
    "nfvopassword": "*****",
    "nfvouername": "*****",
    "tenantname": "admin",
    "token_id": "Qe3mVpzv1ARbZ4Yvp3vFKmyfyvalGSs2",
    "type": "OSM",
    "version": "5"
  }
}

```

```

curl -XGET http://10.100.129.104:8000/api/nfvo/98e1b60e-7d2f-491e-9341-3b464243c408 | python -m json.tool

```

```

{
  "_id": "98e1b60e-7d2f-491e-9341-3b464243c408",
  "config": {
    "network": "flat",
    "network-mode": "provider",
    "wim-sap": "wim-name-1"
  },
  "created_at": 1559818152.5329616,
  "description": "This is a real NFVO to be added",
  "location": "rack-5, NCSRD",
  "name": "nfvo-real-osm",
  "nfvoip": "10.200.64.55",
  "nfvopassword": "*****",
  "nfvouername": "*****",
  "tenantname": "admin",
  "token_id": "Qe3mVpzv1ARbZ4Yvp3vFKmyfyvalGSs2",
  "type": "OSM",
  "version": "5"
}

```

9.1.3. WIM

9.1.3.1. Endpoints

| Method | URL | Data | Description |
|--------|-------------------|-----------------|---|
| GET | /api/wim | - | Returns a list of WIM and their details |
| GET | /api/wim/<wim_id> | - | Returns the details of specific WIM |
| POST | /api/wim | WIM Config file | Adds a new WIM instance – Responses with the WIM ID |

| | | | |
|---------------|-------------------|-----------------|-------------------------------|
| DELETE | /api/wim/<wim_id> | - | Removes a WIM instance |
| PUT | /api/wim/<wim_id> | WIM Config file | Updates a registered WIM info |

9.1.3.2. Examples

```
echo $(curl -X POST -H "Content-Type: application/json" -d @wim.json
http://10.100.129.104:8000/api/wim)
b655bb43-bafc-4271-b3f4-9265f3d89711
```

```
curl -XGET http://10.100.129.104:8000/api/wim | python -m json.tool
```

```
[
  {
    "_id": "b655bb43-bafc-4271-b3f4-9265f3d89711",
    "created_at": 1559818349.5475748,
    "description": "Athens Platform WIM",
    "name": "tnm-wim",
    "url": "http://10.30.0.190:8000"
  }
]
```

```
curl -XGET http://10.100.129.104:8000/api/wim/b655bb43-bafc-4271-b3f4-9265f3d89711
| python -m json.tool
```

```
{
  "_id": "b655bb43-bafc-4271-b3f4-9265f3d89711",
  "created_at": 1559818349.5475748,
  "description": "Athens Platform WIM",
  "name": "tnm-wim",
  "url": "http://10.30.0.190:8000"
}
```

9.1.4. VIM

9.1.4.1. Endpoints

| Method | URL | Data | Description |
|---------------|-------------------|-----------------|---|
| GET | /api/vim | - | Returns a list of VIM and their details |
| GET | /api/vim/<vim_id> | - | Returns the details of specific VIM |
| POST | /api/vim | VIM Config file | Adds a new VIM instance Responses with the VIM ID |
| DELETE | /api/vim/<vim_id> | - | Removes a VIM instance |
| PUT | /api/vim/<vim_id> | VIM Config file | Updates a registered VIM info |

9.1.4.2. Examples

```
echo $(curl -X POST -H "Content-Type: application/json" -d @vim.json
http://10.100.129.104:8000/api/vim)
```

```
c74dbbb7-c092-487c-9fbc-87b3cc96d86e
```

```
curl -XGET http://10.100.129.104:8000/api/vim | python -m json.tool
```

```
[
  {
    "_id": "c74dbbb7-c092-487c-9fbc-87b3cc96d86e",
    "admin_project_name": "admin",
    "auth_url": "http://10.200.64.2:5000/v3/",
    "created_at": 1559818553.571605,
    "description": "This is a real VIM to be added",
    "location": "core",
    "name": "core_vim_1",
    "nap": [
      {
        "dpid": "NonSDN",
        "id": "eno2"
      },
      {
        "dpid": "NonSDN",
        "id": "eno3"
      },
      {
        "dpid": "Celtics",
        "id": "eno4",
        "ports": [
          "11",
          "12"
        ]
      }
    ],
    "password": "*****",
    "type": "openstack",
    "username": "*****",
    "version": "Queens"
  }
]
```

9.1.5. EMS

9.1.5.1. Endpoints

| Method | URL | Data | Description |
|--------|-------------------|------|---|
| GET | /api/ems | - | Returns a list of EMS and their details |
| GET | /api/ems/<ems_id> | - | Returns the details of specific EMS |

| | | | |
|---------------|-------------------|-----------------|---|
| POST | /api/ems | EMS Config file | Adds a new EMS instance - Responses with the EMS ID |
| DELETE | /api/ems/<ems_id> | - | Removes an EMS instance |
| PUT | /api/ems/<ems_id> | EMS Config file | Updates a registered EMS info |

9.1.5.2. Examples

```
echo $(curl -X POST -H "Content-Type: application/json" -d @ems.json
http://10.100.129.104:8000/api/ems)
```

```
e14c9416-e3f1-45e9-9fdc-d397d4aecb8f
```

```
curl -XGET http://10.100.129.104:8000/api/ems | python -m json.tool
```

```
[
  {
    "_id": "e14c9416-e3f1-45e9-9fdc-d397d4aecb8f",
    "created_at": 1559818753.2394924,
    "description": "Athens Platform EMS",
    "name": "athens_ems",
    "url": "http://10.30.0.171:5000"
  }
]
```

```
curl -XGET http://10.100.129.104:8000/api/ems/e14c9416-e3f1-45e9-9fdc-d397d4aecb8f
| python -m json.tool
```

```
[
  {
    "_id": "e14c9416-e3f1-45e9-9fdc-d397d4aecb8f",
    "created_at": 1559818753.2394924,
    "description": "Athens Platform EMS",
    "name": "athens_ems",
    "url": "http://10.30.0.171:5000"
  }
]
```

9.1.6. Registered End-to-End Services

9.1.6.1. Endpoints

| Method | URL | Data | Description |
|------------|---------------------------|------|---|
| GET | /api/service | - | Returns a list of registered services and their details |
| GET | /api/service/<service_id> | - | Returns the details of specific service |

| | | | |
|---------------|---------------------------|---------------------|--|
| POST | /api/service | service Config file | Add new registered services - Responses with a list of IDs |
| DELETE | /api/service/<service_id> | - | Removes a registered service |
| PUT | /api/service/<service_id> | service Config file | Updates a registered service |

9.1.6.2. Examples

```
echo $(curl -X POST -H "Content-Type: application/json" -d @service.json
http://10.100.129.104:8000/api/service)
```

```
['a05ee3e5-e19a-4b43-8d3c-b0e70bfd4380', 'c772c4c9-99e4-4a17-bf62-1e5ee44cfff8']
```

```
curl -XGET http://10.100.129.104:8000/api/service | python -m json.tool
```

```
[
  {
    "_id": "a05ee3e5-e19a-4b43-8d3c-b0e70bfd4380",
    "created_at": 1559817804.16811,
    "name": "NCSRD_embb",
    "ns": [
      {
        "location": "core",
        "name": "vepc",
        "naps": [
          "ens1",
          "ens2"
        ]
      },
      {
        "location": "core",
        "name": "firewall",
        "naps": [
          "ens1"
        ],
        "ns_graph": [
          [
            {
              "nap": 1,
              "ns": "end_point"
            },
            {
              "nap": "ens1",
              "ns": "firewall"
            },
            {
              "nap": "ens1",
              "ns": "vepc"
            },
            {
              "nap": "ens2",
              "ns": "vepc"
            }
          ]
        ]
      }
    ]
  }
]
```

```

        {
            "nap": 2,
            "ns": "end_point"
        }
    ]
}
],
"type": "embb"
},
{
    "_id": "c772c4c9-99e4-4a17-bf62-1e5ee44cfff8",
    "created_at": 1559817804.1681461,
    "name": "NCSRD_urllc",
    "ns": [
        {
            "location": "minilab",
            "name": "vepc",
            "naps": [
                "ens1",
                "ens2"
            ]
        }
    ],
    "type": "urllc"
}
]

```

9.2. MANO Component Registration Templates

9.2.1. New VIM Template

```

{
  "name": <new_VIM_name>,
  "auth_url": <new_VIM_endpoint_url>,
  "username": <admin_username>,
  "password": <admin_password>,
  "admin_project_name": <admin_project_name>,    # Optional
  "location": <nfvi_location>,
  "type": <type>,
  "version": <version>,                          # Optional
  "description": <description>,                  # Optional
  "config":          # Optional
    "security_groups": <sec_group_policy>
}

```

9.2.2. New NFVO Template

```

{
  "name": <new_nfvo_name>,
  "nfvoip": <new_nfvo_endpoint_url>,
  "nfvousername": <admin_username>,
  "nfvopassword": <admin_password>,
  "tenantname": <tenant_name>,                  # Optional
}

```

```
"location": <location>,      # Optional
"type": <nfvo_type>,
"version": <type>,
"description": <description>,  # Optional
"config": {      # Optional
  "network": "flat",
  "network-mode": "provider",
  "wim-sap": "wim-name-1"
}
```

9.2.3. New WIM Template

```
{
  "name": <wim_name>,
  "description": <wim_description>,
  "url": <wim_url>,
  "username": <admin_username>,
  "password": <admin_password>
}
```

9.2.4. New EMS Template

```
{
  "name": <ems_name>,
  "description": <ems_description>,
  "url": <ems_url>,
  "username": <admin_username>,
  "password": <admin_password>
}
```

9.2.5. New Monitoring System Template

```
{
  "name": <mon_name>,
  "description": <mon_description>,
  "url": <mon_url>,
  "username": <admin_username>,
  "password": <admin_password>
}
```

9.3. Versioning Policy

Katana Slice Manager follows the Semantic Versioning 2.0.0³ policy: Given a version number MAJOR.MINOR.PATCH, increment the:

- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards-compatible manner, and
- PATCH version when you make backwards-compatible bug fixes.

³ <https://semver.org/spec/v2.0.0.html>

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

9.4. Network Slice Template

This section describes the defined scheme of the NST for Release A of the Slice Manager.

```
---
nsi:
  id: <string>
  name: <string>
  short-name: <string>
  description: <string>
  type: <embb || urllic || miot>
  nsd-ref:
    - id: <string>
      name: <string>
      short-name: <string>
  wim-ref:
    endpoint-segment:
      - service_id: <string>
        service_name: <string>
        ip: <string>
      - service_id: <string>
        service_name: <string>
    topology: ELAN
    bidirectional: <boolean>
    link_params:
      bw: <integer>
      latency: <float>
  radio-ref:
    location: <string>
```