5Genesis

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

[H2020 - Grant Agreement No. 815178]

Deliverable D3.4

Slice Management (Release B)

Editor T. Anagnostopoulos (NCSRD)

Contributors NCSRD (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 POLITECNICA DE VALENCIA)

Version 1.0

Date Mar 31st, 2021

Distribution PUBLIC (PU)

List of Authors

NCSRD	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. Frascola	
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	

© 5GENESIS Consortium

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.

© 5GENESIS Consortium

Copyright

Copyright © 2021 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	UK
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	Germany
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 D3.4	T. Anagnostopoulos, G. Xilouris (NCSRD)	12.04.2021

© 5GENESIS Consortium

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

© 5GENESIS Consortium

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
PromQL	Prometheus Query Language
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

© 5GENESIS Consortium

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 evolution of design, specifications and implementation of the 5GENESIS Network Slice Manager and more specifically the Release B 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. 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.

The document discusses directly the evolutions, deltas and newly introduced components, acknowledging the fact the Deliverable 3.1 is still valid for the reader to check the use cases, requirements and the background that lead to this particular design. The newly introduced components tackle three different functionalities were needed i.e. i) the internal communication between the components, that is tackled by the implementation of the message bus; ii) per slice and overall slice monitoring exporting tackled by the Prometheus node exporter and grafana dashboard implementation and iii) the capability for slice reconfiguration by policies exposed by the slice optimisation component.

The next part of the document, discusses the enhcancements that were implemented in a series of components contributing to the overall stability of the Slice Manager. These enhancements include, the Northbound API, the slice lifecycle management, the slice mapping and monitoring, and the adaptation layer for the communication with the MANO elements. Moreover the document discusses the new features that are available in this release as a concequence of the aforementioned enhancements. We are highlighting the ability to support concurrent slicing besides slice monitoring and optimisation as already mentioned. Special mention has to be made to the full support of CI/CD pipelines both for the development and for the deployment of the Slice Manager, allowing faster adoption by the community, even for standalone usage outside 5GENESIS ecosystem and initial purpose. 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). Furthermore, the functionalities and purpose of the northbound interfaces of the slice manager (NBI) are documented, providing information wrt their implementation. The aforementioned interfaces are supporting REST APIs and swagger documentation is available along with the software release. In this context the GSMA Generic Slice Template that is supported is discussed in this document. Finally the document provides a quick walkthrough of Slice Manager workflow for installation, configuration and Slice lifecycle operations.

© 5GENESIS Consortium

Table of Contents

© 5GENESIS Consortium

© 5GENESIS Consortium

List of Figures

© 5GENESIS Consortium

List of Tables

© 5GENESIS Consortium

1. INTRODUCTION

1.1. Purpose of the Document

The role of Network Slicing in 5G ecosystems and also beyond, is constantly strengthen as more and more functionalities and configuration are allowed at the Radio Access systems in order to support concurrent slicing support. Presently network slicing support across the 5G infrastructure is not uniformly supported, meaning that different technologies provide different levels of slicing support. Moreover in the radio access domain the support is still depedant on the vendor and market availability of the equipment.

5GENESIS introduces the use of the Slice Manager (SM) right between the Testing Automation and the MANO elements in order to facilitate the deployment of vertical services over the 5G infrastructure and allow the process to also be part of the experimentation procedure. The 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 B 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 also implements (in Release B) additional features related to the network slice provisioning and service deployment, policy framework and per slice monitoring. 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 as it is formed in this final release of the Slice Manager. This document is building upon Release A deliverable D3.3 [1] and its contributions should be considered as a delta from the Release A, discussing the additions and evolutions/enhancements since then. In summary it discusses the

- newly introduced functional components, part of the Slice Manager architecture such as the Slice Optimisation, Promethus exporter for monitoring and the message broker.
- enhancements of original functional components that essentially touch multiple functionalities
- new features that emerge out of the compination of the evolutions above.

1.1.1. Document Dependencies

This document is based on specifications, requirements and assumptions as discussed in the initial specifications of 5GENESIS 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 aforementioned information is already captured in the Release A Slice Management deliverable

© 5GENESIS Consortium

D3.3 [1]. Moreover additional information is analyses from other WP3 deliverables that are released along with this one describing evolution in Release B.

Table 1 Document dependencies

id	Document title	Relevance
D2.1 [2]	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.
D3.3 [1]	Slice Management	This document describes the Rel.A of the slice manager specifications. Is used as a base for the evolution in Rel.B described in this deliverable.
D3.5 [3] / D3.6 [4]	Monitoring and analytics	This document provides information relates to the definition of the Slice Instance monitoring that is provided by the Slice Manager and collected for analytics purposes.
D4.X	Platorm Deliverables	Provide information on the infrastructure and MANO elements available in each 5GENESIS platform.

1.2. Structure of the Document

The document is strucuctured as follows. Chapter 2 discusse the new and enhanced components of the Slice Manager under Rel.B. Chapter 3 presents the Generic Slicing Template that is used to accept Slice requests form the 5GENESIS Experiment Lifecycle Manager (ELCM). Chapter 4 provides a the Slice Manager user guide. 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 codebase is open source, this document serves also as an introduction and architecture primer of the 5GENESIS Slice Manager.

© 5GENESIS Consortium

2. RELEASE B OF 5GENESIS SLICE MANAGER

For completeness purposes, as well as for the reader to have a better overview of the 5GENESIS Slice Manager, the first part of this chapter provides a high-level overview of Release A of the Slice Manager. Subsequently, the chapter introduces Release B of the component, including the refined internal architecture, the new services that were implemented as part of the software stack, and the advancements in the functions that were included in Release A.

2.1. Release A Overview

The Release A of the Slice Manager is described in *D3.3 – Slice Management Rel. A* [1] that was submitted on Month 15 of the project. It has been successfully integrated into all 5GENESIS platforms and it was utilized during the second experimentation cycle, as presented in *D6.2 - Trials and experimentation (cycle 2)* [5].

Figure 1 presents an overview of the building blocks that comprise Release A of the 5GENESIS Slice Manager. This first release implements the core functionalities that are required for performing the main lifecycle management operations on network slices. In addition, it provides all necessary interfaces and communication mechanisms for interconnecting with the other components that were part of the 5GENESIS architecture during the second experimentation cycle. More specifically, Release A of the Slice Manager was successfully integrated with following components:

Table 2: Integrated 5GENESIS Components in Release A

Component	Release	Interface
ELCM	Release A	REST
Open Source MANO	Five, Six	REST
OpenStack	Queens, Stein	REST, Socket
ODL WIM	Release A	Kafka Message Bus
Athens EMS	Release A	REST

The integration process of these components was part of the WP5 activities and is described in *D5.1 – System level tests and verification* [6] that was submitted on Month 18.

© 5GENESIS Consortium

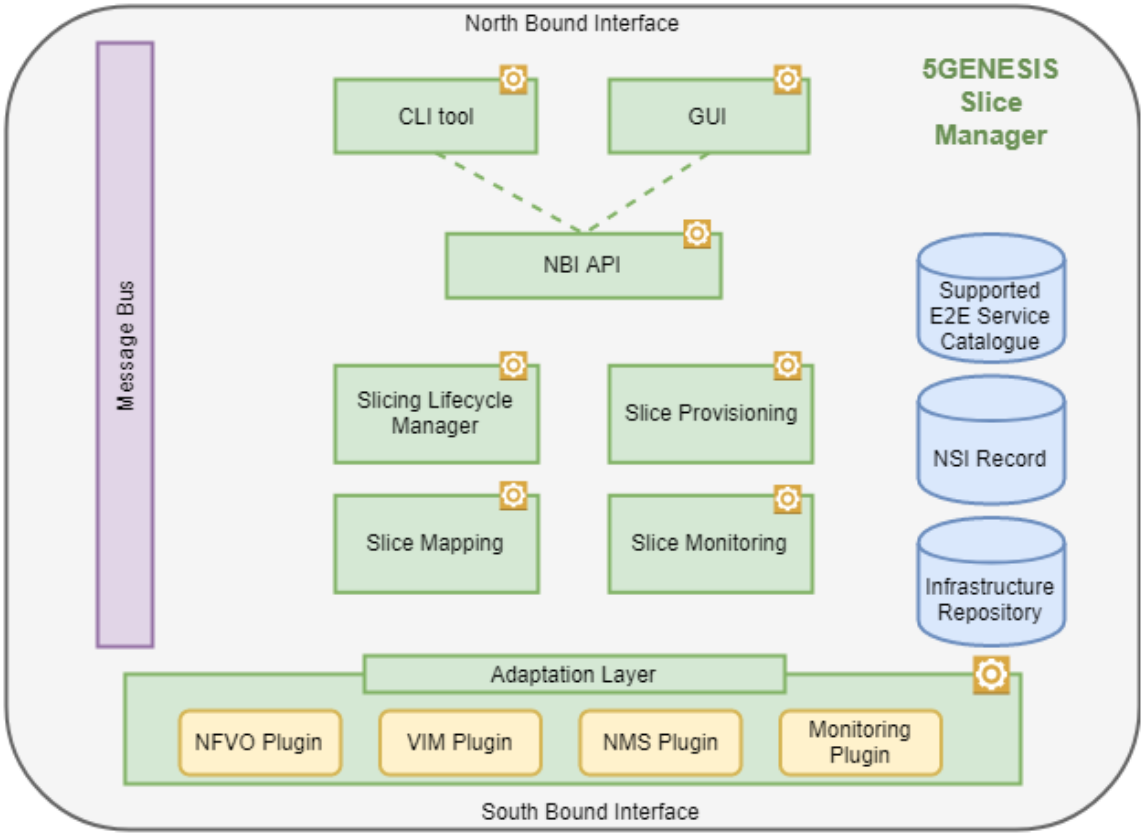


Figure 1: Architecture of the Slice Manager Release A

Release A of the 5GENESIS Slice Manager was published as an open-source software tool in May 2020, under the 5GENESIS group on Github (https://github.com/5genesis/katana-slice_manager/tree/Release_A).

2.2. Introduction to Release B

Release B of the Slice Manager introduces some new modules that, in conjunction with advancements in those modules that were already implemented as part of Release A, offer an enhanced set of functionalities related to the slice management processes. A refined architectural diagram has been created, including all the new modules that comprise Release B. The following sections present the updated Slice Manager architecture, the new modules that have been developed and integrated into the software stack, and the evolution of the existing modules. It seems noteworthy to mention that, as happened with Release A, Release B of the Slice Manager was also developed and published as an open-source software tool under the 5GENESIS group on Github[1].

[1] https://github.com/5genesis/katana-slice_manager/tree/Release_B

© 5GENESIS Consortium

2.3. Release B Architecture

Following the same approach as Release A, Release B of the Slice Manager is built as a mesh of microservices integrated and working collectively to offer slice management services. Each microservice is a module running as a Docker container, forming the Slice Manager's software stack. Figure 2 depicts the Slice Manager's refined architectural diagram, while Figure 3 presents a snapshot of the instantiated Docker container.

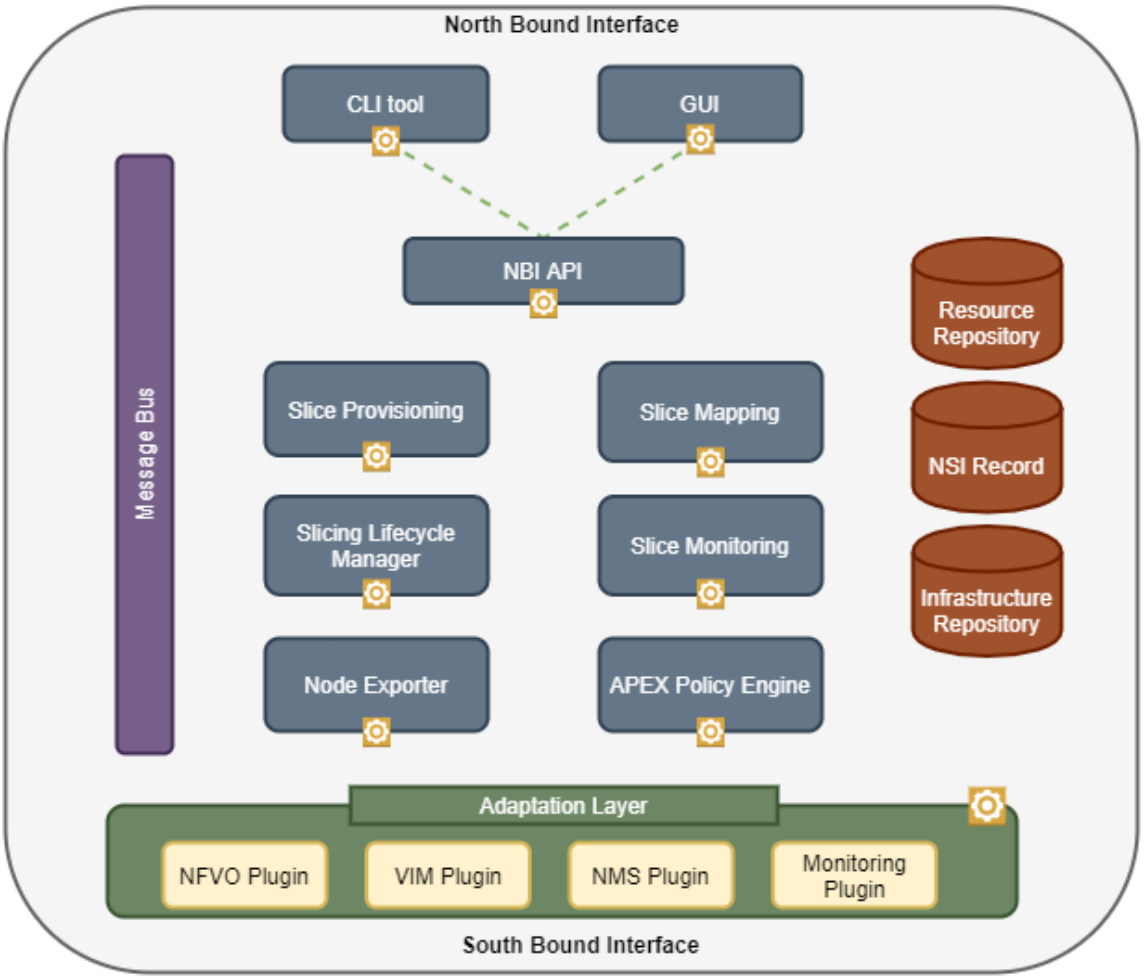


Figure 2: Slice Manager Release B Architecture

© 5GENESIS Consortium

```
IMAGE                                  PORTS                                          NAMES
katana-nfv_mon                         0.0.0.0:8002->8002/tcp                         katana-nfv_mon
swaggerapi/swagger-ui                  80/tcp, 0.0.0.0:8001->8080/tcp                 katana-swagger
katana-mngr                                                                           katana-mngr
katana-cli                                                                            katana-cli
grafana/grafana:7.1.1                  0.0.0.0:3000->3000/tcp                         katana-grafana
katana-nbi                             0.0.0.0:8000->8000/tcp                         katana-nbi
confluentinc/cp-enterprise-kafka:5.4.2 0.0.0.0:9092->9092/tcp                         katana-kafka
confluentinc/cp-zookeeper:5.4.2        2888/tcp, 0.0.0.0:2181->2181/tcp, 3888/tcp     katana-zookeeper
mongo:4.0.5                            27017/tcp                                      katana-mongo
prom/prometheus:v2.19.1                0.0.0.0:9090->9090/tcp                         katana-prometheus
```

Figure 3: Slice Manager Software Stack

2.4. New modules of Release B

Two new modules have been developed and integrated into the Slice Manager's software stack, namely the Prometheus Node Exporter and the APEX Policy Engine.

2.4.1. Prometheus Node Exporter

Prometheus Node Exporter is a service that runs inside a Docker container that is part of the Slice Manager software stack. This service is a collection of multiple system threads that are executed concurrently to periodically gather metrics from the underlying components or the Slice Manager itself. These metrics are used to feed the Slice Manager's monitoring module, using the Prometheus server for collecting the available data and Grafana for visualizing it.

Prometheus Node Exporter collects metrics regarding the number and status of the instantiated slices from other modules that run as the Slice Manager's internal services. This process allows the monitoring module to offer an overview of the instantiated and terminated slices at any time. In addition to that, the Node Exporter also collects information regarding the health status of the virtual network services that are part of every created slice. For this purpose, a dedicated thread is created for every new slice, utilizing the available APIs of the NFVO to gather information regarding the instantiated virtual network services. This process allows the Slice Manager to ensure that all of the underlying virtual services that have been created for the purposes of a slice are properly instantiated and running.

2.4.2. Message Broker

A dedicated message broker enables the communication among the various internal Slice Manager services. Apache Kafka is utilized for implementing the message broker services. Kafka[2] is an open-source distributed event streaming platform that allows software applications to exchange information in real-time in the form of streams of events.

Kafka runs as a cluster of one or more distributednodes, called the brokers. It requires a synchronization service to keep track of the status and manage its nodes. Release B of the Slice

[2] https://kafka.apache.org

© 5GENESIS Consortium

manager introduces these services as internal microservices of the software stack. More specifically, a Kafka broker runs in a Docker container is responsible for handling the streams of events between the other internal services. A Zookeeper instance (also running in a Docker container) is responsible for synchronizing the Kafka cluster.

Slice Manager's internal services can act as data producers or data consumers. Producers are those applications that publish (write) events to Kafka, and consumers are those that subscribe to (read and process) these events. Producers and consumers are fully decoupled and agnostic of each other. The messages that various applications exchange are organized and stored in Kafka topics. Slice Manager creates and maintains topics for each communication pipeline that must be implemented between the internal services.

2.4.3. Slice Optimization Module

The Slice Optimisation Module is designed to optimise the slice management during runtime. The core of this module is the APEX policy engine[3], which allows for flexible run-time adaptations and optimizations. APEX presents a strong tool for automated decision making, being able to handle adaptive policies, i.e. policies that can modify their behavior based on system and network conditions, including decision making at runtime rather than at policy definition time and the ability to use context information that was not provided in the incoming event or request. The functionality of APEX was described in detail in Deliverable D3.1 – Management and Orchestration [7]. We are focusing here on the integration of APEX into the Slice Manager (SM) component, as well as an updated set of proposed use cases.

The APEX engine has two main interfaces:

- An input interface to receive events: also known as ingress interface or consumer, receiving (consuming) events commonly named **triggers**. In our implementation, these triggers come from the Slice Monitoring Module.
- An output interface to publish produced events, also known as egress interface or producer, sending (publishing) events commonly named **actions** or action events. These actions are consumed by Slicing Lifecycle Manager, triggering a day-2 configuration action, depending on the type of action that must be applied.

In our implementation, we will use a Kafka bus for communicating and consuming the triggers and actions between the different modules.

One other important part of the Slice Optimization Module is the actual policy, which contains different tasks activated depending on the context of the triggers. Examples of such policy tasks are: deciding whether a Network Service must be restarted or updating a Network Slice placement based on current conditions of the network.

Similar to the other modules, the APEX engine and the policy elements (as mandated by each optimization use case) have been packaged as a runnable Docker image exposing relevant ports for a number of interfaces, e.g., for alert capturing and action output, as depicted in Figure 4.

[3] https://docs.onap.org/projects/onap-policy-parent/en/latest/apex/APEX-Introduction.html

© 5GENESIS Consortium

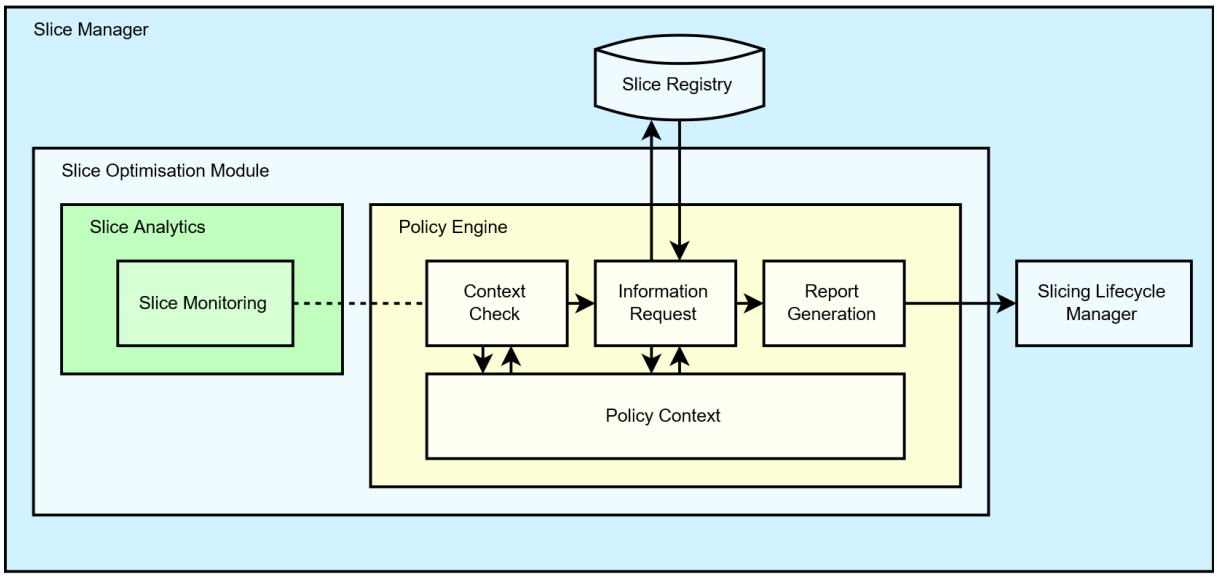


Figure 4: Interaction between slice optimization and slicing LCM modules

2.5. Advancements in existing modules of Release A

In addition to the new modules that have been introduced in Release B, many of the existing modules have been evolved to offer improved services and new functionalities. This section describes the advancements in the existing modules of the Slice Manager.

2.5.1. North Bound Interface, CLI tool, GUI

As described in D3.3, the North Bound Interface module enables a set of APIs that allow other components and users to communicate with the Slice Manager. In addition to the NBI module, two lightweight modules, namely the CLI Tool and the GUI, can be used for interacting with the Slice Manager. All these services have been further developed and upgraded to support the new features of Release B. More specifically, the newly defined APIs are listed below:

- **`/api/bootstrap`**: Configure Slice Manager using a single configuration file
- **`/api/ns`**: List all the available virtual network services and the resource requirements of each one
- **`/api/policy`**: Add, modify or remove an external policy engine component
- **`/api/resources`**: List all the available resources of the underlying infrastructure

For each new endpoint, the respective functionality has been added to the CLI and the GUI tools.

2.5.2. Slice Lifecycle Management

The main feature that has been added to the Slice Lifecycle Management (LCM) module in Release B is the ability to perform day-2 configurations on created slices. More specifically, a user can now modify a slice, either by changing various parameters of the NSSIs that comprise

© 5GENESIS Consortium

the slice or by scaling up/down some of the instantiated virtual services that are also part of the slice.

2.5.3. Slice Mapping

The slice mapping module has been modified to enable NSSIs to be shared among slices that run concurrently. When Slice Manager receives a request for a new slice, based on the definition of the available NSSIs (whether they can be used by multiple slices at the same time or not) and the requirements of the slices, this module decides if the new slice can also use an NSSI that is already part of another running slice. You can read more about the feature of sharing NSSIs between multiple concurrent slices in section 2.6.2.

2.5.4. Slice Monitoring

The Slice Monitoring module in Release B enables monitoring, visualization, and alerting services responsible for tracking the status of components and services that are part of the instantiated slices and the Slice Manager itself. It comprises a Prometheus server and a Grafana web application running in Docker containers as part of the overall Slice Manager software stack.

Prometheus is utilized for collecting and organizing metrics from both physical and virtual components and services of the underlying infrastructure that have been created and configured as part of a deployed slice. These metrics are stored locally in the Prometheus server in a dedicated time-series database. Subsequently, Prometheus runs rules over those measured data points to create newly aggregated data or trigger alerts when specified conditions are observed. These alerts are handled by the Prometheus Alertmanager, a service running alongside the Prometheus server that is responsible for routing the alerts to the correct receiver, such as the Slice Optimization module, as depicted in Figure 4. In addition to the services mentioned above, Prometheus provides a functional query language called PromQL (Prometheus Query Language) that allows the users to select and aggregate time-series data in real-time via a specific API. The result of an expression can either be shown as a graph or viewed as tabular data in Prometheus' expression browser.

Grafana is an open-source analytics and visualization web application. It provides charts, graphs, and alerts when connected to supported data sources organized in fully customizable dashboards. Grafana supports querying Prometheus to use the collected metrics as an input data source.

Section 2.6.1 presents more details regarding the monitoring features that are supported by the Slice Manager monitoring module.

2.5.5. Adaptation Layer

For Release B of the Slice Manager, more plugins have been integrated into the adaptation layer, allowing the Slice Manager to interact with all the components comprising the MANO layer of the 5GENESIS platforms. Table 3 summarizes the versions of the MANO layer components that have been tested and integrated with the Slice Manager's current Release.

© 5GENESIS Consortium

Table 3: Supported MANO Components

MANO Component	Type	Release
NFVO	Open Source MANO	Five - Eight
VIM	Openstack	Queens, Rocky, Stein, Train, Ussuri
WIM	ODL WIM	Release A, Release B
EMS	Athens Platform EMS	Release A, Release B

2.6. New Features in Release B

The new modules that were introduced in the Slice Manager software stack and the advancements in the modules that were part of Release A have built a new set of features and functionalities offered by the Slice Manager. These features aim to provide an enhanced slice management service and fulfill some of the use cases, as they were described in *D2.1 – Requirements of the Facility* [2]. This section presents the new features that are available in release B of the Slice Manager.

2.6.1. Slice Monitoring

Slice monitoring involves tracking and analyzing network components and functions that the Slice Manager has instantiated and configured as part of a network slice. This process provides insights on the performance while ensures continuous uptime and good health of the services realized by every network slice. Platform administrators can utilize the slice monitoring feature to quickly detect networking failures and determine in real-time whether the platform is running optimally. Release B of the 5GENESIS Slice Manager capitalizes on Prometheus and Grafana to create a toolkit that offers monitoring, visualization, and alerting capabilities. The tools are packaged in Docker containers and delivered as part of the Slice Manager microservices architecture, as described in 2.5.4.

Slice Manager exploits Prometheus as a time-series database. It collects, organizes, and stores metrics from targets by scraping HTTP endpoints. These targets are either physical or virtual components and services of the underlying infrastructure that have been instantiated and configured to be part of a deployed slice. Prometheus offers a file-based service discovery mechanism, which allows adding new targets dynamically in a JSON file, along with metadata about those data. Every component of the MANO layer that the platform administrator adds to the Slice Manager is configured to be a new Prometheus target if the component supports it. This feature enables Slice Manager to collect monitoring information from various domains that are part of each slice. More specifically, Prometheus can gather data from the following sources:

- WIM: It collects metrics related to the network traffic from components like routers, switches, and firewalls. Tracking various aspects of the network traffic within a slice

© 5GENESIS Consortium

(e.g., bandwidth utilization, loss, delay, etc.) allows to quickly identify connection failures or issues such as traffic bottlenecks that limit data flow.

- NFVO: NFVO monitoring mechanisms collect data related to the health and the performance of the instantiated VNFs in the context of a deployed network slice. Slice Manager's Prometheus can concentrate these metrics for all slices in a centralized database.
- VIM: Prometheus can collect data from the VIMs related to the instantiated VMs, such as CPU, memory, and disk utilization. For VIM metrics to be collected, the VIM should support a Telemetry system.

Prometheus exposes the collected data through an HTTP REST API, allowing other monitoring or visualization systems to collect the data. This functionality allows the Slice Manager Prometheus to be part of a hierarchical Prometheus federation. Prometheus also realizes a graphical web user interface, which the administrator can use to add new targets, configure rules for aggregating scraped data into new time series, or run queries against the collected data. Moreover Prometheus provides a Web UI for directly creating queries and plots as well as defining rules and filters for alerts.

Alerting rules is another valuable feature of Prometheus that the Slice Manager utilizes. This feature allows the platform administrator to define alert conditions based on Prometheus expression language. Whenever the alerting criteria are met for one or more elements at a given point in time, Prometheus sends notifications about the firing alerts to external services.

In addition to the process of scrapping the data, Slice Manager utilizes Grafana to visualize the collected metrics stored in the Prometheus database. Grafana allows the creation of multiple dynamic and reusable dashboards that include custom visualizations based on predefined templates. Slice Manager exploits this capability to create a dedicated dashboard for every deployed slice, concentrating and visualizing the metrics related to each one. This approach allows the Slice Manager administrator to efficiently keep track of the services and components that are instantiated and configured as part of a slice. When a network slice is terminated, Slice Manager dynamically deletes the respective Grafana dashboard. Figure 5 provides a snapshot of a Grafana dashboard that the Slice Manager created for a deployed slice. In this example, Grafana visualizes metrics collected from i) the NFVO related to the health and status of the instantiated VNFs, ii) the WIM related to the network traffic in the slice, and iii) the VIM related to the performance of the created VMs.

© 5GENESIS Consortium

Figure 5: Grafana Slice Monitoring

Furthermore, Grafana creates a Dashboard that provides an overview of the deployed slices and their status. This Dashboard is based on a static configuration and is used as the Grafana home page dashboard. The figure below depicts a snapshot of the home dashboard.

© 5GENESIS Consortium

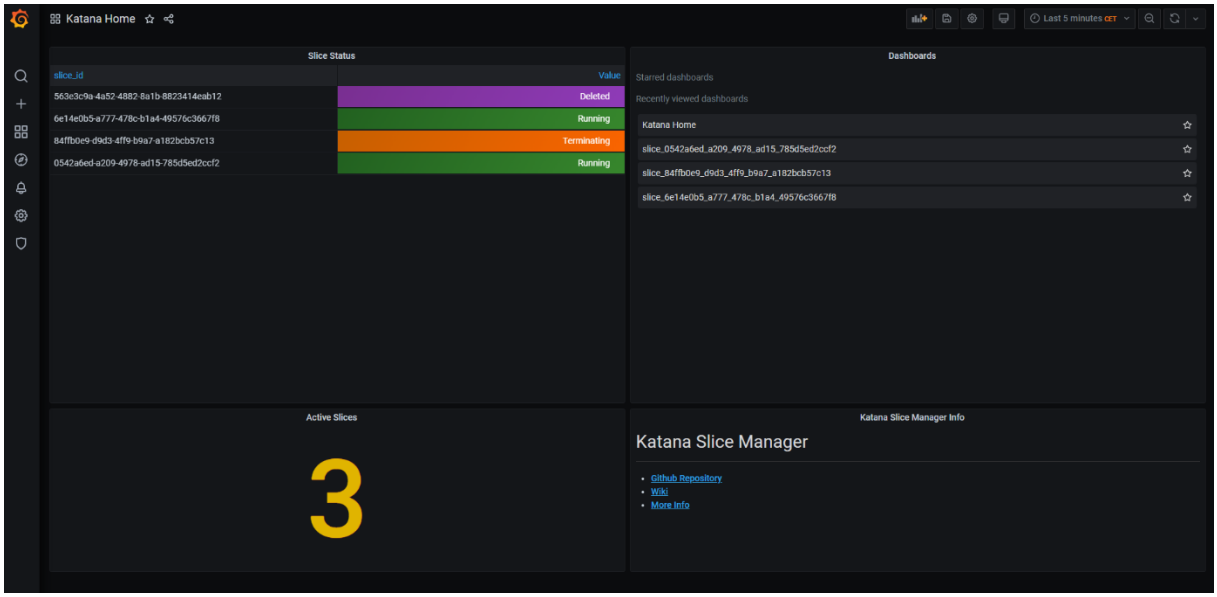


Figure 6: Grafana Home Dashboard

2.6.2. Concurrent slices with shared NSSIs

The ability to share physical or virtual network resources among multiple slices is a significant aspect of the Slice Management procedure. 5GENESIS Slice Manager allows a new slice to share a specific part of the infrastructure that is already part of another slice. More specifically, as depicted in Figure 7, two or more slices can concurrently use an instantiated and configured NSSI. In this example, Slice 1, which comprises three NSSIs, has already been deployed on the platform. Slice 2, which also comprises three NSSIs, is then deployed, using one of the NSSIs that has already been realized as part of Slice 1. This process allows the Slice Manager to manage the resources of the underlying infrastructure more effectively.

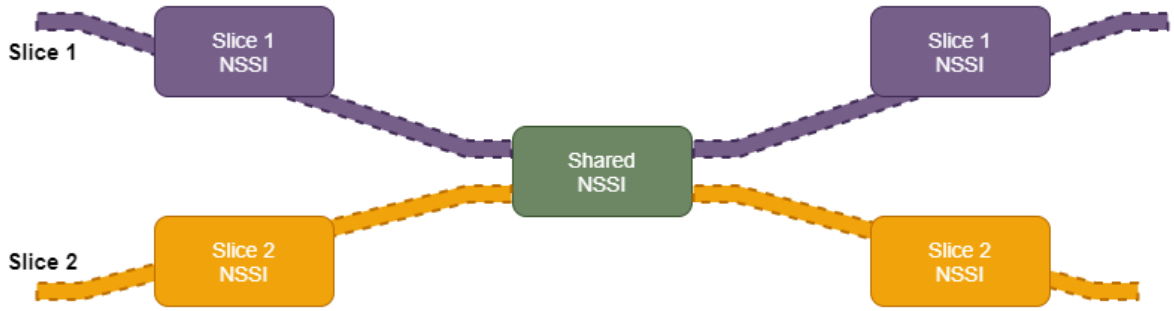


Figure 7: Shared NSSI among two network slices

For an NSSI to be shared among multiple network slices, some specific criteria must be met. Firstly, the NSSI must explicitly specify that multiple slices can use it in parallel. Secondly, as an additional condition, each slice's NEST request must define an isolation level that permits sharing the NSSIs with other slices. Figure 8 presents a diagram depicting the steps that the

© 5GENESIS Consortium

Slice Manager follows to determine whether an NSSI will be shared among multiple slices or not.

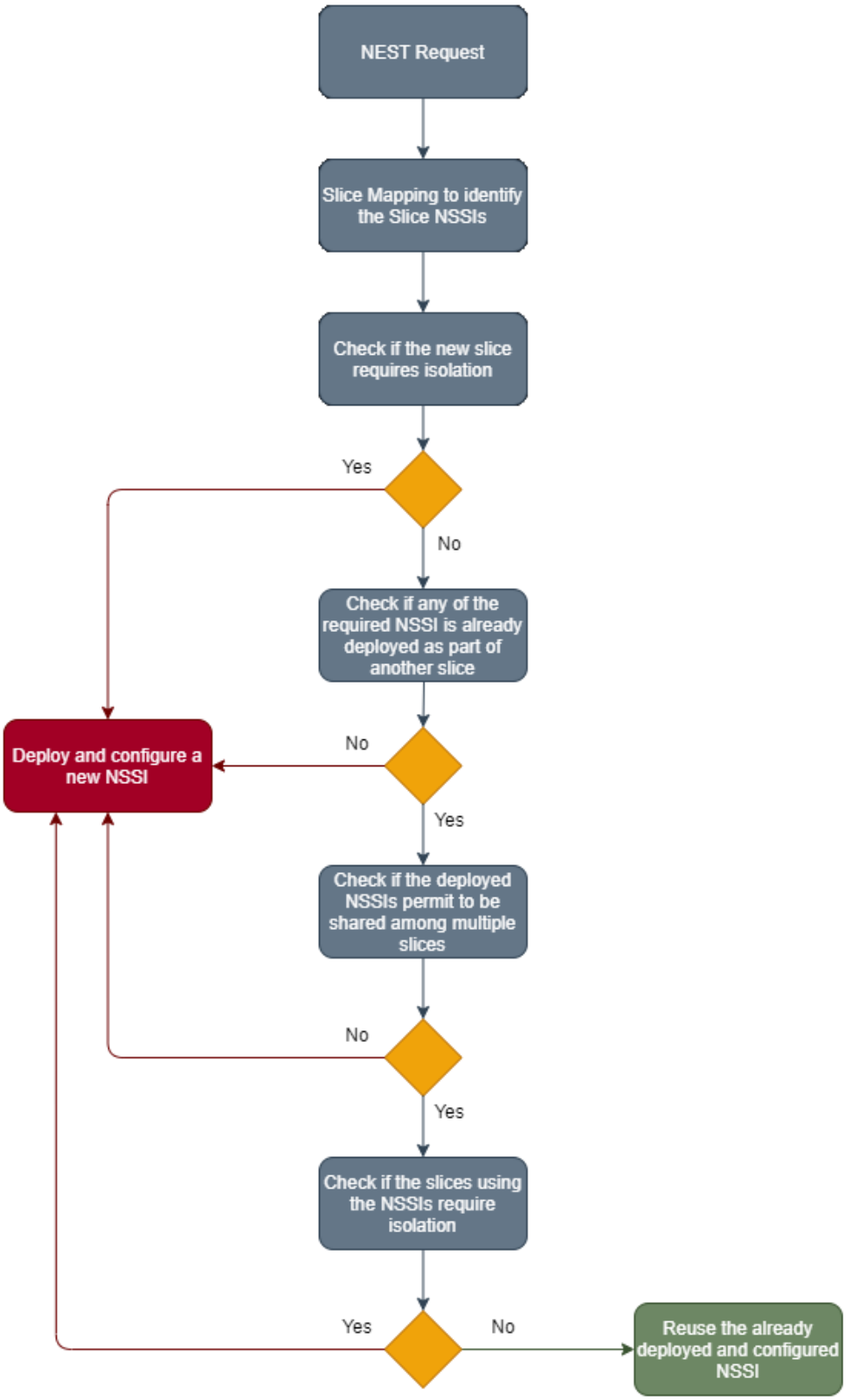


Figure 8 Decision Diagram for shared NSSIs

© 5GENESIS Consortium

2.6.3. Slice Optimization

Release B of the Slice Manager supports the interconnection with one or more policy engines for enabling network slice optimization mechanisms. The first option that has been implemented is the integrated Slice Optimization Module. This module allows for the slice optimization process, driven by the APEX policy engine. This process is triggered by the Prometheus Alert Manager's alerts, which runs as part of the Slice Monitoring module. A use case scenario for this module is when the Slice Monitoring module detects a stopped or errored virtual network service that is part of a running slice and sends an alert to the APEX policy engine. APEX processes the incoming request and recommends appropriate actions, such as restarting the network service. The UML diagram below presents the flow of actions that corresponds to such a use case scenario.

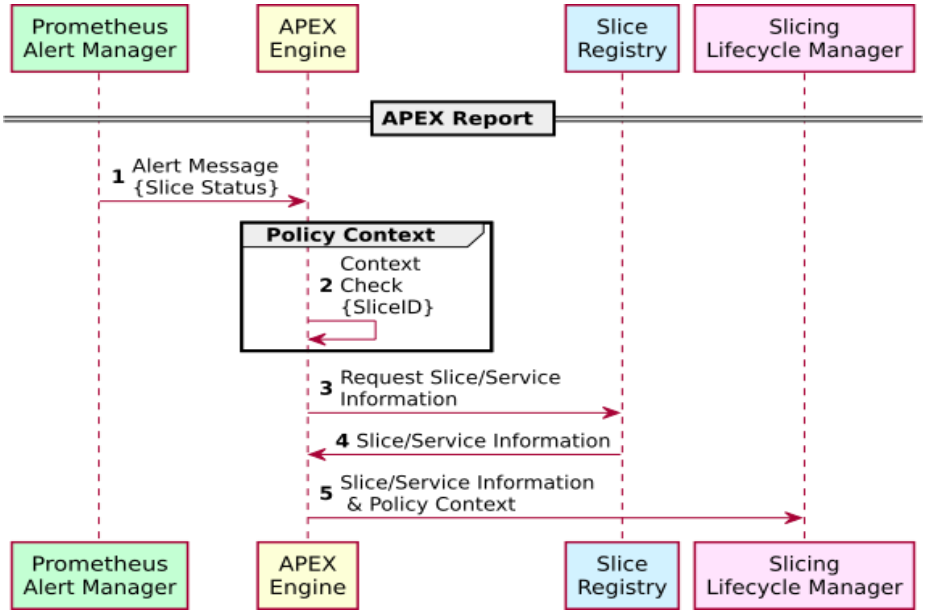


Figure 9: Use Case diagram for APEX driven Slice Optimization

2.6.4. CI/CD Pipelines

Release B of the 5GENESIS Slice Manager was developed following the best practices of the Continuous Integration (CI) – Continuous Deployment (CD) methodology. CI process ensures that when developers, as members of a team, commit their code changes into a central repository, they do not introduce any errors or bugs. For this purpose, CI automates the integration of code changes from multiple contributors into a single software project. A shorter feedback loop means having more iterations. Each iteration introduces automated builds and unit tests on the latest code changes to immediately detect any errors, improving the overall software quality.

The CI/CD pipelines are implemented as a series of steps that the CI server has to execute. A typical CI pipeline includes the following steps:

© 5GENESIS Consortium

i. A Slice Manager developer implements changes to the source code on their local systems. Then, they commit the changes and push the new source code to the remote code repository.
ii. CI server receives a notification from the source code repository when a push event occurs.
iii. CI server pulls the latest version of the source code, builds the Slice Manager services, and runs the defined tests.
iv. If the tests are completed successfully, the CI server assigns a build tag to the version of the code it just built and pushed the produced artifacts to a registry.
v. If the tests reveal any errors or bugs, the team fixes the issues and runs a new iteration.

Several scripts have been created and included in the Slice Manager source code repository to realize the CI pipelines in the Slice Manager development. These scripts enable the automated build, testing, and deployment of the Slice Manager services, allowing CI servers to validate their proper functionality.

Jenkins is currently the most popular and widely used CI server. In Jenkins, CI pipelines are described in specific files called *Jenkinsfiles*, which are usually included in the source code repository and are pulled by Jenkins whenever a new build is triggered. Jenkinsfile allows developers to implement pipelines in code formats. Slice Manager includes two Jenkinsfiles that any Jenkins server can utilize to implement two different workflows, namely the development and the deployment workflows.

2.6.4.1. Development Workflow

In the development workflow, developers commit source code changes and push their code to the central source code repository. This event automatically triggers Jenkins to pull, compile, build, and test the latest version from the repository's specific feature branch. If the unit tests finish successfully, Jenkins uses Docker to create Docker images, which then pushes to the Docker Registry with an appropriate tag. The development pipeline is depicted in Figure 10.

© 5GENESIS Consortium

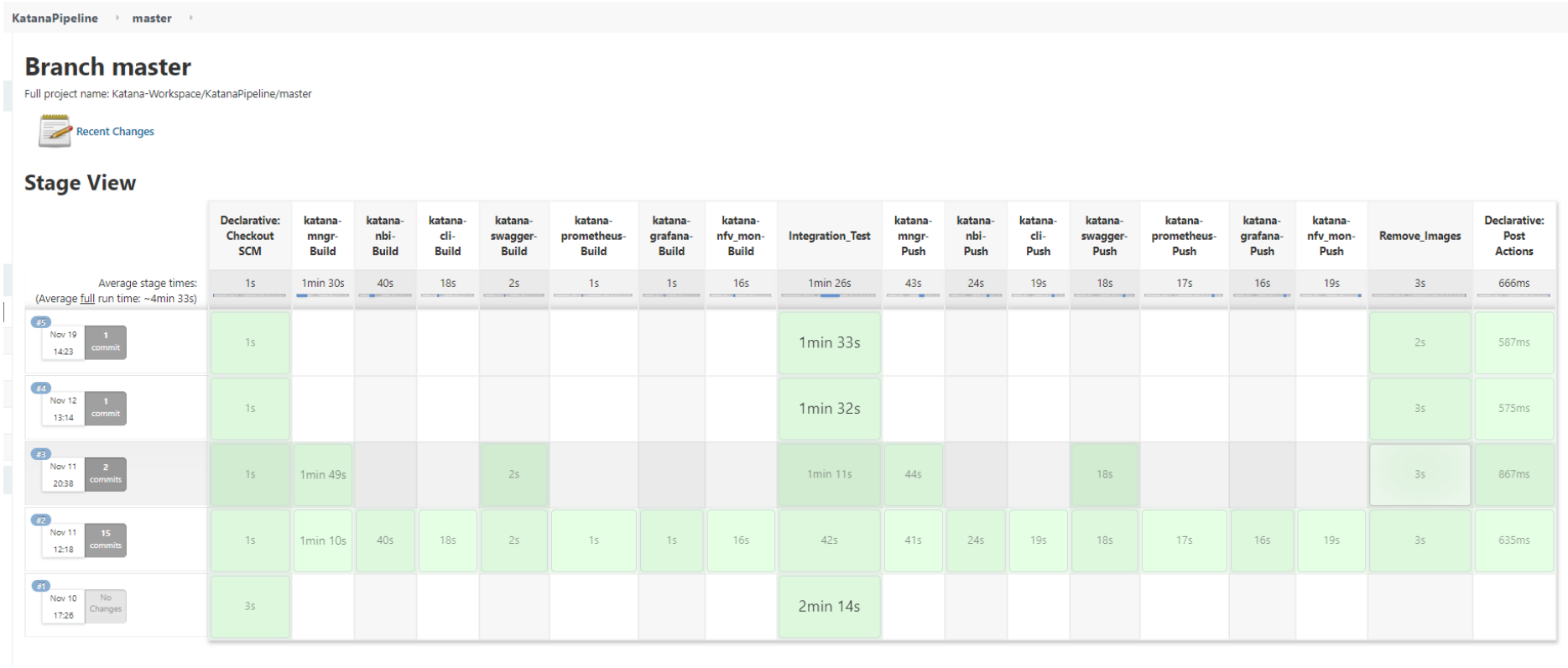


Figure 10: CI/CD Development Pipeline

© 5GENESIS Consortium

2.6.4.2. Deployment Workflow

The deployment workflow is executed when the development of a new feature is completed and a new stable version of the Slice Manager is ready to be released. In this workflow, a developer creates a new tag with the appropriate versioning number and pushes it to the central repository. This action automatically triggers Jenkins to execute a new pipeline, which downloads the latest docker image from the Docker registry and deploys the new version of the application on the defined operational environment of the 5GENESIS platforms. The deployment pipeline is depicted in Figure 11.

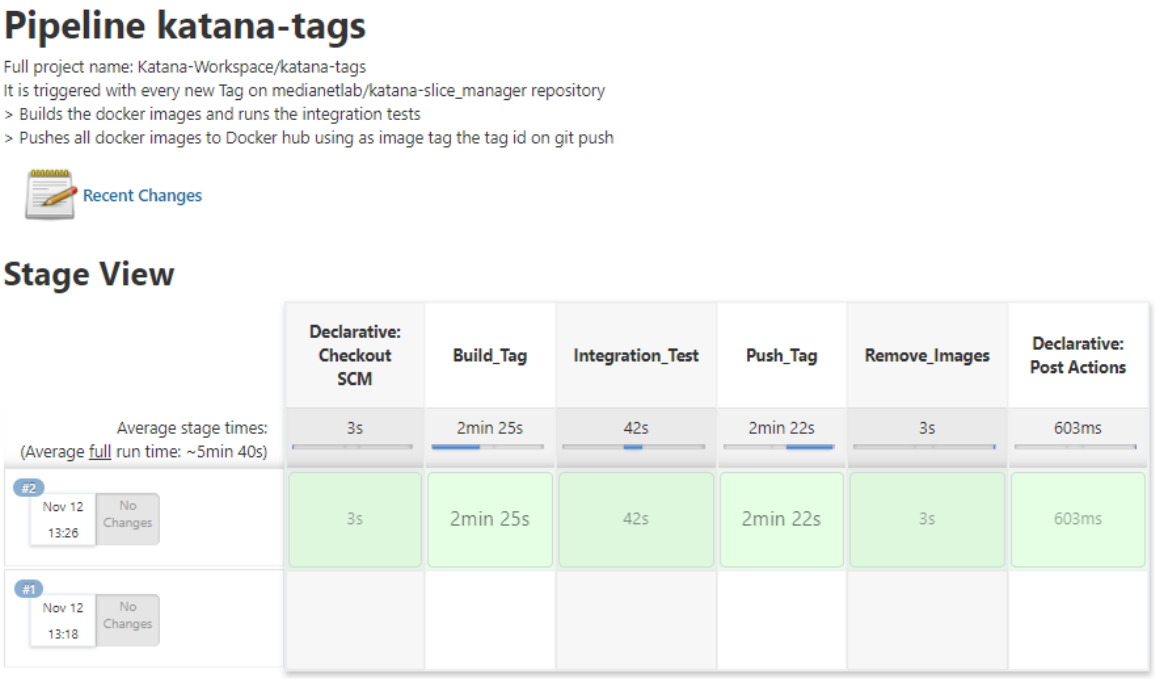


Figure 11: CI/CD Deployment Pipeline

© 5GENESIS Consortium

3. NETWORK SLICE TEMPLATE

The Network Slice Template (NEST) is used for describing the parameters of a network slice. A NEST must complement a slice creation request that a user sends to the Slice Manager to deploy a new slice. This chapter describes the NEST scheme defined and used by the Slice Manager in Release B, while Annex 7.2 presents the complete JSON Schema of the NEST.

3.1. Network Slice Preparation

The Network Slice Template that describes the parameters of network slices in the context of 5GENESIS follows the specifications of *NG.116 – Network Slice Template v2.0* [8], defined by the GSMA. This document aims to assist network slice providers in mapping network slices' use cases into generic attributes. For this purpose, GSMA defines the Generic Network Slice Template (GST), a set of attributes that characterize a network slice type. GST is generic and is not tied to any specific network deployment.

The NEtwork Slice Template (NEST) is a GST filled with values. The defined attributes and their values are assigned to fulfill a given set of requirements derived from a network slice use case. This process is depicted in the diagram of Figure 12, where a Network Slice Customer (NSC) provides the requirements for a specific use case to the Network Slice Provider (NSP). The requirements are mapped into the attributes of the GST with appropriate values generating a NEST.

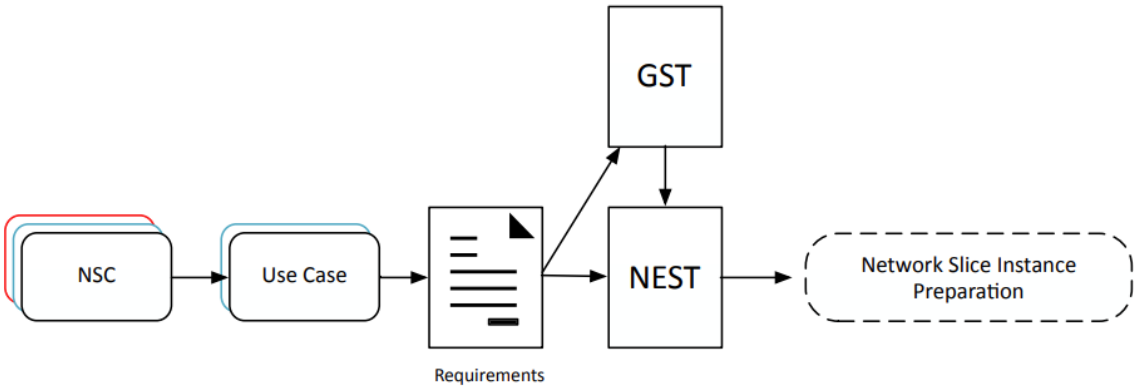


Figure 12: Network slice creation request

The NEST is an input to the network slice (instance) preparation performed by the Slice Manager. The user sends a NEST to Katana Slice Manager along with a slice creation request. The Slice Mapping process then parses the NST, which, combined with the supported network functions supported by the underlying infrastructure, defines the NSSIs that shall be deployed as part of the slice.

© 5GENESIS Consortium

3.2. 5GENESIS GST

The GST that the Slice Manager uses for describing network slices in the context of 5GENESIS implements the most significant attributes that GSMA defines in [8]. We can split the GST into three sections:

- Base Slice Descriptor: This section includes the main parameters that the Slice Manager utilizes for the Slice Mapping process, such as location, bandwidth requirements, isolation level, QoS, etc. This process selects the appropriate NSSIs that will be used for constructing the new slice.
- Vertical Services Descriptor: This section describes network services that are not part of the slice's core service but are defined by the Network Slice Customer, such as firewalls, caches, proxies, etc. These services can be either physical or virtual and must be supported by the underlying platform infrastructure.
- Test Descriptor: This section specifies a series of tests that the Slice Manager must run against the newly deployed slice. These tests can validate various aspects of the slice, such as the service's performance and reliability.

While the Base Slice Descriptor is a mandatory section for a NEST that is used for the deployment of a new slice, the other two sections are optional and might be omitted. Slice Manager allows the administrator to onboard descriptors for each section before the slice creation phase. Moreover, it creates an endpoint that returns a list with all the onboarded descriptors. The onboarded descriptors can be referenced in a NEST using a UUID on a slice creation event. Finally, a NEST can reference a previously onboarded descriptor and also define some parameters for that section. In this scenario, the Slice Manager uses the referenced descriptor as a base, replacing the parameters that are also defined in the NEST.

© 5GENESIS Consortium

4. SLICE MANAGER RELEASE B USER GUIDE

This chapter provides guidance to administrators on how to install and configure Release B of the 5GENESIS Slice Manager on their system. Although this guide uses the built-in CLI tool for interacting with the Slice Manager, the same functionalities can be realized using the REST endpoints of the North Bound Interface. You can refer to Annex 7.1 for a complete list of the available REST endpoints.

4.1. Slice Manager Installation

5GENESIS Slice Manager is implemented as a collection of multiple microservices that are running in Docker containers. These microservices comprise the Slice Manager software stack. The communication among the various Docker containers is enabled with a dedicated Docker network, allowing the services to reach each other by referencing the respective container names. The slice manager's installation requires a virtual or physical Linux host that supports Docker and docker-compose services. The host's minimum resource requirements to support the Slice Manager services are at least four gigabytes of memory and two CPU cores.

The first step is to download the source code from the 5GENESIS Github group and checkout to Release B:

```
git clone https://github.com/5genesis/katana-slice_manager.git
git checkout Release_B
```

The next step is to build the Docker images for all the Slice Manager services and start the Docker containers. These tasks can be completed with the use of the "*installation*" and "*deployment*" scripts, respectively:

```
bash install.sh
bash start.sh
```

Note that the installation script requires root user privileges. The start script can receive the following options:

- -m or --monitoring: This option deploys the Docker containers that implement the Slice Manager monitoring module, namely Prometheus server, Prometheus node exporter, Grafana, and APEX Policy engine.
- -p or --publish: This option exposes the Slice Manager message broker, allowing it to be reachable from external services.
- -g or --graphical-ui: This option deploys the Docker containers that implement the Slice Manager Graphical User Interface.

After these steps, inspecting the Docker containers should return a similar list to the one depicted in the figure below.

© 5GENESIS Consortium

```
IMAGE                                PORTS                                          NAMES
katana-nfv_mon                       0.0.0.0:8002->8002/tcp                         katana-nfv_mon
swaggerapi/swagger-ui                80/tcp, 0.0.0.0:8001->8080/tcp                 katana-swagger
katana-mngr                                                                         katana-mngr
katana-cli                                                                          katana-cli
grafana/grafana:7.1.1                0.0.0.0:3000->3000/tcp                         katana-grafana
katana-nbi                           0.0.0.0:8000->8000/tcp                         katana-nbi
confluentinc/cp-enterprise-kafka:5.4.2  0.0.0.0:9092->9092/tcp                      katana-kafka
confluentinc/cp-zookeeper:5.4.2      2888/tcp, 0.0.0.0:2181->2181/tcp, 3888/tcp     katana-zookeeper
mongo:4.0.5                          27017/tcp                                      katana-mongo
prom/prometheus:v2.19.1              0.0.0.0:9090->9090/tcp                         katana-prometheus
```

Figure 13: Slice Manager Docker Containers

To stop the Slice Manager services, the administrator can use the "*stop*" script. Using the -c or –clear option removes the persistent Docker volumes that store the instantiated services' data. In addition to that, the "*uninstall*" script can be used for removing all the Docker resources related to the Slice Manager services, such as images, networks, and volumes. Note that the "*uninstall*" script also requires root privileges:

```
bash stop.sh -c
bash uninstall.sh
```

4.2. Slice Manager Configuration

After successfully installing and deploying the services, the administrator must configure the Slice Manager repositories with the components and network functions available at the platform's MANO and Infrastructure layers. For these tasks, the administrator can use the built-in 5GENESIS CLI tool called *katana* or the REST endpoints available at the Slice Manager NBI. This guide uses the *katana* CLI tool, while the list with the available REST API endpoints is presented in Annex 7.1. The attributes of the platform's components and network functions are described in JSON configuration files that the Slice Manager consumes. Slice Manager's repository contains templates that administrators can use for defining the available components in their platforms, available in the URL: https://github.com/5genesis/katana-slice_manager/tree/Release_B/example_config_files

The Slice Manager administrator must add configuration files with attributes about the following components before deploying the first slice:

- Network Functions: Define the available physical and virtual network functions available in the platform's infrastructure. These functions implement the slice's core 5G service. Each network function comprises an NSSI, and during the slice deployment phase, multiple functions are stitched together to construct the end-to-end service of the slice. A network function can be added with the following command:

  ```
  katana function add -f <function_conf_file.json>
  ```

- VIMs: Define components of the infrastructure layer utilized for hosting virtual services created as part of network slices. Slice Manager creates a new tenant for every new

© 5GENESIS Consortium

slice that is deployed on the platform. A VIM can be added to the Slice Manager with the following command:

```
katana vim add -f <vim_conf_file.json>
```

- NFVOs: Define components of the platform's MANO layer responsible for performing lifecycle operations on the deployed virtual services. An NFVO can be added to the Slice Manager with the following command:

```
katana nfvo add -f <nfvo_conf_file.json>
```

- WIM: Define the platform's MANO layer components responsible for managing the WAN of a deployed slice, setting parameters such as QoS, bandwidth capabilities, delay tolerance, and so forth. A WIM can be added to the Slice Manager with the following command:

```
katana wim add -f <wim_conf_file.json>
```

- EMS: Define the platform's MANO layer components responsible for managing and configuring Physical Device Units that are part of the deployed slices. The physical devices primarily are components of the Radio Access Network of a slice. An EMS can be added to the Slice Manager with the following command:

```
katana ems add -f <ems_conf_file.json>
```

All the commands listed above return a UUID that can be used for identifying the specific component. While having added some network functions, VIMs, and NFVOs to the Slice Manager is a requirement, WIMs and EMSs are optional components for creating and deploying a network slice. If there are no WIMs or EMSs configured in the Slice Manager's repositories when a slice creation request arrives, the Slice Manager deploys the slice without configuring the WAN and the PDUs.

Further than adding the components to the Slice Manager repositories, the administrator can use the following commands to list, inspect, modify or delete these components:

```
katana <component> ls
katana <component> inspect <component_uuid>
katana <component> rm <component_uuid>
katana <component> update <component_uuid> -f <new_configuration_file.json>
```

4.3. Slice Management

When the platform's administrator has configured the Slice Manager, registering all the available components that comprise the platform's MANO and infrastructure layer, he can proceed to the slice creation phase. The initial step is to create the NEST that the Slice Manager will use to create the network slice. Creating the NEST requires the platform administrator to identify the proper NEST values that express and fulfill a given set of requirements to support the network slice use case. The NEST is then provided as an input to the Slice Manager with a new slice deployment request:

© 5GENESIS Consortium

```
katana slice add -f NEST.json
```

The Slice Manager returns a UUID, an identifier for the new slice that the administrator can use for slice querying purposes. The slice creation request goes through the Slice Manager's various modules and internal services, such as the Slice Mapping, Placement, Resource Provisioning, and service activation, before it reaches the running state. This process is extensively described in D3.3. The administrator can use the following commands to list, inspect or delete the deployed network slices with the following commands:

```
katana slice ls
katana slice inspect <slice_uuid>
katana slice rm <slice_uuid>
```

Furthermore, the administrator can get information regarding the deployment time of the slice. The following command will return a JSON file with information regarding the deployment of the slice and each service:

```
katana slice deployment_time <slice_uuid>
```

© 5GENESIS Consortium

5. CONCLUSIONS AND FUTURE WORK

This document presented the activities related to the implementation of the Slice Manager in order to support network slicing within the 5GENESIS framework. It discusses Rel B. interfaces and architectural design of the 5GENESIS Slice Manager. Finally, it highlighted the new features and enhancements from Rel. A. A summary is provided below:

- newly introduced functional components, part of the Slice Manager architecture such as the Slice Optimisation, Promethus exporter for monitoring and the message broker.
- enhancements of original functional components that essentially touch multiple functionalities
- new features that emerge out of the compination of the evolutions above.

The official development period for the 5GENESIS Slice Manager is completed by the release of this document. The Slice Manager will be used in the future both for the realisation of the 5GENESIS Use Cases showcasing and experiments as well as for other activities beyond 5GENESIS project. In the future additional enhancements will be introduced both at the level of supporting newcoming features and capabilities at the Radio Access with respect to slicing as well as extending particular functionalities for the support of QoS, concurrency and other runtime operations.

© 5GENESIS Consortium

6. BIBLIOGRAPHY

[1] 5GENESIS Consortium, "D3.3 Slice management," 2019. [Online]. Available: https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.3_v1.0.pdf. [Accessed June 2020].

[2] 5GENESIS Consoritum, "D2.1 Requirements of the Facility," 2018. [Online]. Available: https://5genesis.eu/wp-content/uploads/2018/11/5GENESIS_D2.1_v1.0.pdf.

[3] 5GENESIS Consortium, “Delvierable D3.5 Monitoring and Analytics,” 2019. [Online]. Available: https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.5_v1.0.pdf. [Accessed 6 2020].

[4] 5GENESIS Consortium, "D3.6 Monitoring and Analytics," [Online]. Available: to appear in 2021. [Accessed 2021].

[5] 5GENESIS Consortium, "The Malaga Platform," 2020. [Online]. Available: https://5genesis.eu/wp-content/uploads/2020/02/5GENESIS_D4.2_v1.0.pdf. [Accessed June 2020].

[6] 5GENESIS Consortium, "The Berlin Platform," 2020. [Online]. Available: https://5genesis.eu/wp-content/uploads/2020/02/5GENESIS_D4.14_v1.0.pdf. [Accessed June 2020].

[7] 5GENESIS Consortium, "D3.1 - Management and Orchestration," September 2019. [Online].

[8] GSMA Association, "GSMA," 16 October 2019. [Online]. Available: https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v2.0.pdf.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

© 5GENESIS Consortium

[13] 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.

[14] 5GENESIS, "D3.15 - Expirement Life Cycle Manager," September 2019. [Online].

[15] Linux Foundation, "ONAP," [Online]. Available: https://wiki.onap.org.

[16] 5GTANGO, "5GTANGO," [Online]. Available: https://wiki.onap.org.

[17] Sonata, "TANGO Slice Manager," [Online]. Available: https://github.com/sonata-nfv/tng-slice-mngr/wiki.

[18] MATILDA, "MATILDA," [Online]. Available: https://www.matilda-5g.eu/.

[19] MATILDA Consortium, "Network and Computing Slice," September 2019. [Online]. Available: https://private.matilda-5g.eu/documents/PublicDownload/487.

[20] MongoDB, [Online]. Available: https://www.mongodb.com/.

[21] "ETSI MEC Specifications," [Online]. Available: https://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing.

[22] 5GENESIS Consortium, "5GENESIS Github," [Online]. Available: https://github.com/5genesis. [Accessed June 2020].

[23] 5GENESIS Consortium, "D3.3 Slice Management," 2020. [Online]. [Accessed 30 3 2021].

[24] 5GENESIS Consortium, "Slice Management," 2020. [Online]. Available: https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.5_v1.0.pdf. [Accessed 30 3 2021].

[25] 5GENESIS Consortium, "The Limassol Platform," 2020. [Online]. Available: https://5genesis.eu/wp-content/uploads/2020/02/5GENESIS_D4.5_v1.0.pdf. [Accessed June 2020].

[26] 5GENESIS Consortium, "The Athens Platform," 2019. [Online]. Available: https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.5_v1.0.pdf. [Accessed June 2020].

[27] 5GENESIS Consortium, "Deliverable D6.1 Trials and experimentation (cycle1)," 2019. [Online]. Available: http://5genesis.eu/wp-content/uploads/2019/12/5GENESIS_D6.1_v2.00.pdf. [Accessed June 2020].

[28] 5GENESIS Consortium, "Management and Orcestration," 2020. [Online]. Available: https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.1_v1.0.pdf. [Accessed June 2020].

[29] 3GPP, "3GPP TS 22.179 Mission Critical Push to Talk (MCPTT); Stage 1," 3GPP, 2019.

© 5GENESIS Consortium

[30] 3GPP, "3GPP TS 37.340 NR; Multi-connectivity; Overall description; Stage-2," 3GPP, Dec 2018.

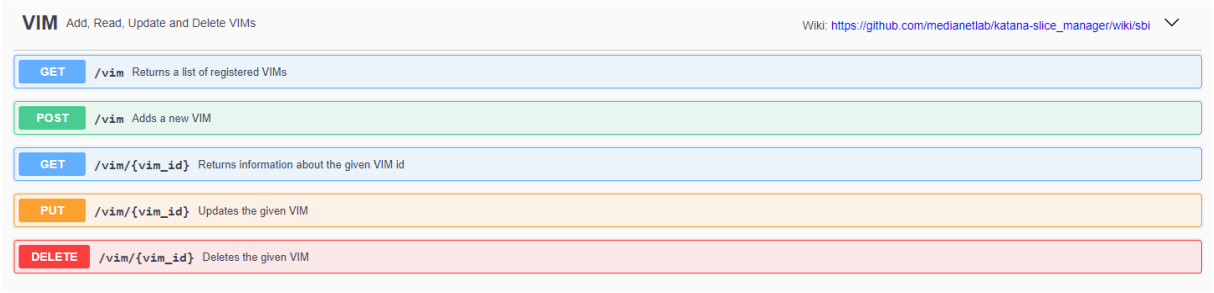
© 5GENESIS Consortium

7. APPENDICES

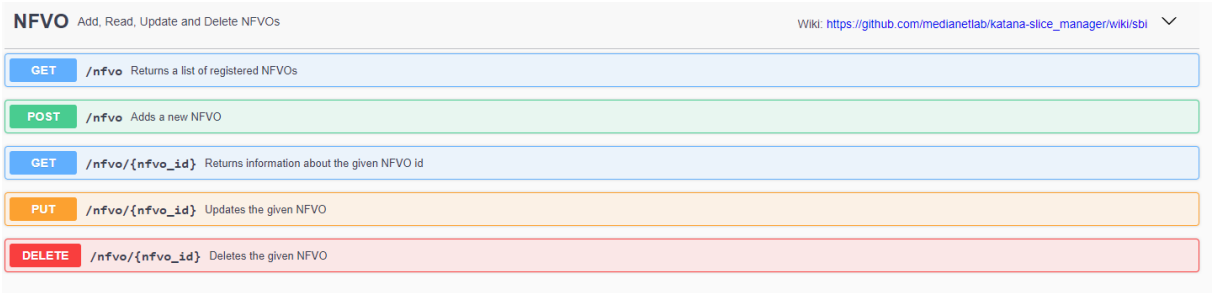
7.1. Swagger API definition

- /api/slice

- /api/vim

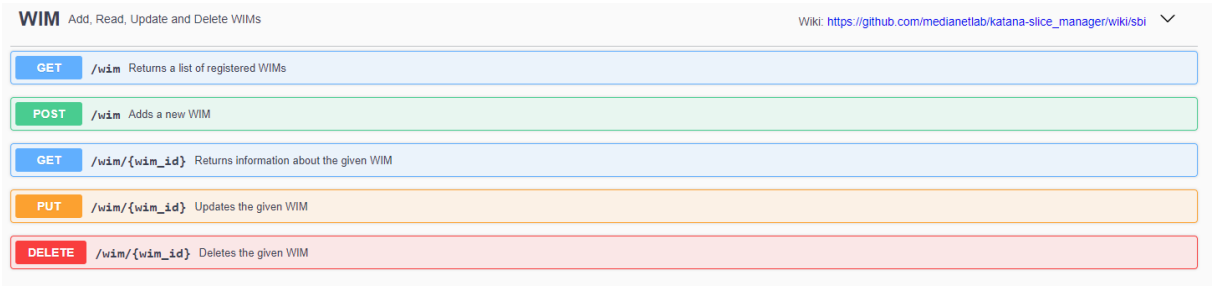


- /api/nfvo

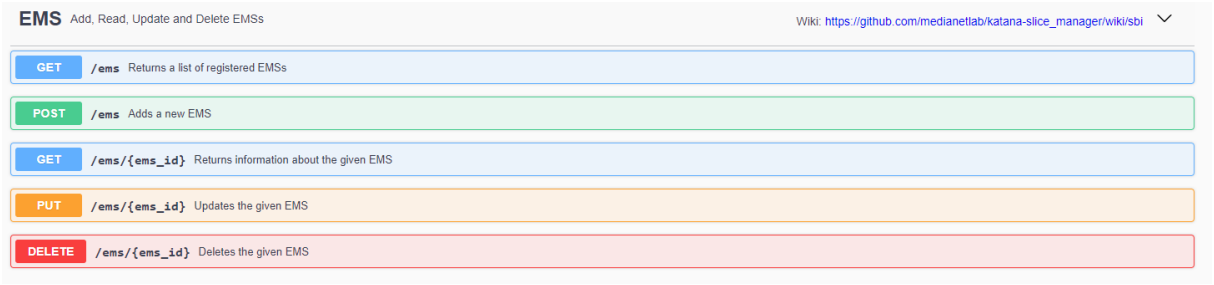


© 5GENESIS Consortium

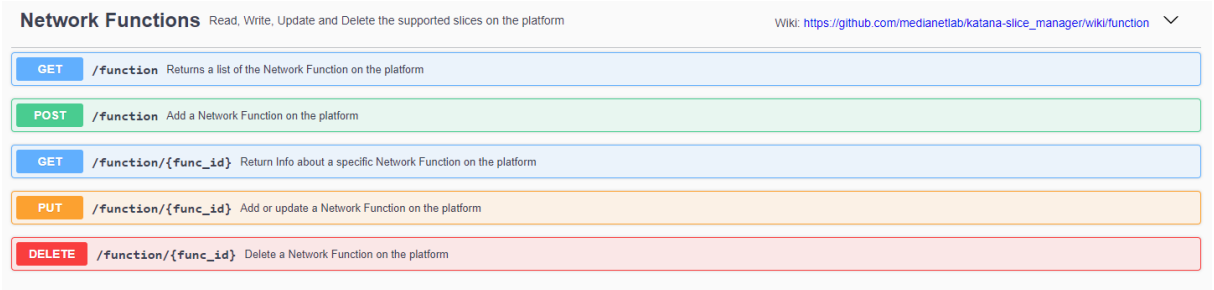
- /api/wim



- /api/ems



- /api/function



7.2. NEST JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",

  "definitions": {
    "base_slice_descriptor":{
      "type": "object",
      "description": "This is the schema for the core \
       part of the new slice",
      "properties": {
        "base_slice_des_id": {
          "type": "string",
          "description": "Id of the slice descriptor \
           which will be used as base for the gst"
        },
```

```
        "base_slice_des_ref": {
          "type": "string",
          "description": "Optional - Reference to an added \
            slice descriptor which will be used as base for the gst"
        },
        "coverage": {
          "type": "array",
          "description": "A list with all the locations that \
            are part of the slice",
          "items":{
            "type": "string",
            "description": " The location name for each site in the slice"
          }
        },
        "delay_tolerance": {
          "type": "boolean",
          "description": "Supported or not supported"
        },
        "deterministic_communication":{
          "type": "object",
          "description": "This attribute defines if the \
            network slice supports deterministic communication \
            for periodic user traffic. Periodic traffic refers \
            to the type of traffic with periodic transmissions.",
          "properties":{
            "availability": {
              "type": "boolean",
              "description": "This parameter describes if the \
                  network slice supports deterministic communication."
            },
            "periodicity": {
              "type": "array",
              "description": "This parameter provides a \
               list of periodicities supported by the network slice.",
              "items": {
                "type": "number",
                "description": "Seconds"
              }
            }
          }
        },
        "network_DL_throughput": {
          "type": "object",
          "description": "The achievable data rate in downlink \
           for the whole network slice (and not per user).",
          "properties": {
            "guaranteed": {
              "type": "number",
              "description": "kbps"
            },
            "maximum": {
              "type": "number",
              "description": "kbps"
            }
          }
        },
        "ue_DL_throughput": {
          "type": "object",
          "description": "This attribute describes the \
           guaranteed data rate supported by the network slice \
           per UE in downlink",
```

© 5GENESIS Consortium

```
          "properties": {
            "guaranteed": {
              "type": "number",
              "description": "kbps"
            },
            "maximum": {
              "type": "number",
              "description": "kbps"
            }
          }
        },
        "network_UL_throughput": {
          "type": "object",
          "description": "The achievable data rate in uplink \
           for the whole network slice (and not per user).",
          "properties": {
            "guaranteed": {
              "type": "number",
              "description": "kbps"
            },
            "maximum": {
              "type": "number",
              "description": "kbps"
            }
          }
        },
        "ue_UL_throughput": {
          "type": "object",
          "description": "This attribute describes the \
           guaranteed data rate supported by the network slice \
           per UE in uplink",
          "properties": {
            "guaranteed": {
              "type": "number",
              "description": "kbps"
            },
            "maximum": {
              "type": "number",
              "description": "kbps"
            }
          }
        },
        "group_communication_support": {
          "type": "number",
          "enum": [0, 1, 2, 3],
          "description": "0: not available 1: Single Cell \
           Point to Multipoint (SCPTM) 2: Broadcast/Multicast \
           3: Broadcast/Multicast + SC-PTM"
        },
        "isolation_level": {
          "type": "object",
          "description": " A network slice instance may be fully or \
           partly, logically and/or physically, isolated from \
           another network slice instance",
          "properties": {
            "isolation": {
              "type": "number",
              "enum": [0, 1, 2, 3],
              "description": "0: No Isolation 1: Physical Isolation \
               2: Logical Isolation 3: Both Isolation"
            }
```

© 5GENESIS Consortium

```
        }
      },
      "mtu": {
        "type": "number",
        "description": "Bytes"
      },
      "mission_critical_support": {
        "type": "object",
        "description": "Mission-critical (MC) leads to a priority of \
         the network slice relative to others, for C-plane and \
         U-plane decisions.",
        "properties": {
          "availability": {
            "type": "boolean"
          },
          "mc_service": {
            "type": "array",
            "description": "This attribute specifies whether or \
             not the network slice supports MC push-to-talk, MC data, \
             MC video, Isolated E-UTRAN Operation for Public Safety \
             or MC interworking.",
            "items": {
              "type": "number",
              "enum": [1, 2, 3, 4, 5],
              "description": "1: MCPTT 2: MCData 3: MCVideo 4: IOPS \
               5: MC interworking"
            }
          }
        }
      },
      "mmtel_support": {
        "type": "boolean",
        "description": "This attribute describes whether the \
         network slice supports IP Multimedia Subsystem (IMS) \
         and Multimedia Telephony Service MMTel."
      },
      "nb_iot": {
        "type": "boolean",
        "description": "This parameter describes whether NB-IoT \
         is supported in the network slice."
      },
      "number_of_connections": {
        "type": "number",
        "description": "This attribute describes the maximum number \
         of concurrent sessions supported by the network slice."
      },
      "number_of_terminals": {
        "type": "number",
        "description": "This attribute describes the maximum number \
         of concurrent terminals supported by the network slice."
      },
      "positional_support": {
        "type": "object",
        "description": "This attribute describes if the network \
         slice provides geo-localization methods or supporting methods.",
        "properties": {
          "availability": {
            "type": "array",
            "description": "Describes if this attribute is provided \
             by the network slice and contains a list of \
             positioning methods provided by the slice.",
```

© 5GENESIS Consortium

```
              "items": {
                "type": "number",
                "enum": [1, 2, 3, 4, 5, 6, 7],
                "description": "1: CID 2: E-CID (LTE and NR) 3: OTDOA \
                 (LTE and NR) 4: RF fingerprinting 5: AECID 6: \
                 Hybrid positioning 7: NET-RTK"
              }
            },
            "frequency": {
              "type": "number",
              "description": "Seconds"
            },
            "accuracy": {
              "type": "number",
              "description": "Meters"
            }
          }
        },
        "radio_spectrum": {
          "type": "array",
          "description": "Defines the radio spectrum supported \
           by the network slice.",
          "items": {
            "type": "string",
            "description": "This attribute simply tells which \
             frequencies can be used to access the network slice.
             Example: n1, n77, n38"
          }
        },
        "simultaneous_nsi": {
          "type": "number",
          "enum": [0, 1, 2, 3, 4, 5],
          "description": "0: Can be used with any network slice \
           1: Can be used with network slices with same SST value \
           2: Can be used with any network slice with same SD value
           3: Cannot be used with another network slice \
           4-15: operator defined class"
        },
        "qos": {
          "type": "array",
          "description": "This attribute defines all the QoS \
           relevant parameters supported by the network slice, based \
           on 3GPP defined standard values (5QIs)",
          "items": {
            "type": "object",
            "description": "Refers to 5QI defined in \
https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.
aspx?specificationId=3144",
            "properties": {
              "qi": {
                "type": "number",
                "description": "Based on the table of defined 5QI by 3GPP"
              },
              "resource_type": {
                "type": "number",
                "enum": [0, 1, 2],
                "description": "0: GBR (Mission Critical Video user \
                 plane) 1: Delay critical GBR (Intelligent \
                 Transport Systems) 2: Non-GBR (Voice, AR)"
              },
              "priority_level": {
```

© 5GENESIS Consortium

```
                "type": "number",
                "description": "Associated with 5G QoS \
                 characteristics indicates a priority in \
                 scheduling resources among QoS Flows."
              },
              "packet_delay_budget": {
                "type": "number",
                "description": "The Packet Delay Budget (PDB) defines \
                 an upper bound for the time that a packet may be \
                 delayed between the UE and the UPF [Seconds]."
              },
              "packet_error_rate": {
                "type": "number",
                "description": "The Packet Error Rate (PER) defines \
                 an upper bound for the rate of packets that are \
                 not successfully delivered by the corresponding \
                 receiver [percentage]."
              },
              "jitter": {
                "type": "number",
                "description": "Jitter is defined as a variation in \
                 the delay of received packets [Seconds]."
              },
              "max_packet_loss_rate": {
                "type": "number",
                "description": " the maximum rate for lost packets of \
                 the QoS flow that can be tolerated in the uplink (UL) \
                 and downlink (DL) direction [percentage."
              }
            }
          }
        },
        "nonIP_traffic": {
          "type": "boolean"
        },
        "device_velocity": {
          "type": "number",
          "enum": [1, 2, 3, 4],
          "description": "1: Stationary: 0 km/h 2: Pedestrian: 0 km/h to \
           10 km/h 3: Vehicular: 10 km/h to 120 km/h 4: High \
           speed vehicular: 120 km/h to 500 km/h"
        },
        "terminal_density": {
          "type": "number",
          "description": "maximum number of connected and/or \
           accessible devices per unit area (per km2) supported by \
           the network slice [Number/km^2]"
        }
      }
    },
    "service_descriptor":{
      "type": "object",
      "description": "This is the schema for the Service Descriptor part \
       of slice IM",
      "properties": {
        "ns_list": {
          "type": "array",
          "description": "List of the NSD to be instantiated alongside \
           the slice",
          "items": {
            "type": "object",
```

© 5GENESIS Consortium

```
            "description": "A NS",
            "properties":{
              "nfvo-id":{
                "type": "string",
                "description": "The NFVO that will manage the life \
                 cycle of the NS"
              },
              "nsd-id": {
                "type": "string",
                "description": "The NSD id as defined on the NFVO"
              },
              "ns-name": {
                "type": "string",
                "description": "The name of the NS"
              },
              "placement": {
                "type": "number",
                "enum": [0, 1],
                "description": "1: Core, 2: Edge"
              }
            }
          }
        }
      }
    },
    "test_descriptor":{
      "type": "object",
      "description": "This is the schema for the Test Descriptor \
       part of slice IM",
      "properties": {
        "probe_list": {
          "type": "array",
          "description": "A list of probe ids to be included in the slice",
          "items": {
            "type": "string"
          }
        },
        "performance_monitoring": {
          "type": "object",
          "description": "This attribute provides the capability to \
           monitor KQIs and KPIs.",
          "properties": {
            "availability": {
              "type": "array",
              "description": "List of KQIs and KPIs available \
               for monitoring",
              "items": {
                "type": "number",
                "enum": [1, 2, 3],
                "description": "1: Throughput 2: Latency 3: \
                 Service Request Success Rate"
              }
            },
            "frequency": {
              "type": "number",
              "description": "Seconds"
            }
          }
        },
        "performance_prediction": {
          "type": "object",
```

```
          "description": "This attribute provides the capability to \
           predict KQIs and KPIs.",
          "properties": {
            "availability": {
              "type": "array",
              "description": "List of KQIs and KPIs available \
               for monitoring",
              "items": {
                "type": "number",
                "enum": [1, 2, 3],
                "description": "1: Throughput 2: Latency 3: Service Request
Success Rate"
              }
            },
            "frequency": {
              "type": "number",
              "description": "Seconds"
            }
          }
        }
      }
    }
  },


  "type": "object",
  "properties": {
    "base_slice_descriptor": { "$ref": "#/definitions/base_slice_descriptor"
},
    "service_descriptor": { "$ref": "#/definitions/service_descriptor" },
    "test_descriptor": { "$ref": "#/definitions/test_descriptor" }
  },
  "required": ["base_slice_descriptor"]
}
```

© 5GENESIS Consortium