A Holistic, Innovative Framework for the Design, Development and Orchestration of 5G-ready Applications and Network Services over Sliced Programmable Infrastructure

Co-funded by the Horizon 2020 Framework Programme of the European Union

DELIVERABLE D6.13

VALIDATION RESULTS, PERFORMANCE EVALUATION AND ADOPTION GUIDELINES – SECOND DEMONSTRATION AND EVALUATION PHASE

Due Date of Delivery:	M38 *Mx* (31/07/2020 *dd/mm/yyyy*)
Actual Date of Delivery:	16/10/2020 *dd/mm/yyyy*
Workpackage:	**WP6 – MATILDA Industrial Demonstrators and Performance Evaluation**
Type of the Deliverable:	R
Dissemination level:	PU
Editors:	COSM, CNIT, UBITECH
Version:	1.0

Call:
H2020-ICT-2016-2

Type of Action:
IA

Project Acronym:
MATILDA

Project ID:
761898

Duration:
38 months

Start Date:
01/06/2017 *dd/mm/yyyy*

Project Coordinator:
Name: Franco Davoli
Phone: +39 010 353 2732
Fax: +39 010 353 2154
e-mail: franco.davoli@cnit.it

Technical Coordinator:
Name: Panagiotis Gouvas
Phone: +30 216 5000 503
Fax: +30 216 5000 599
e-mail: pgouvas@ubitech.eu

List of Authors	
CNIT	CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI
R. Bruschi, C. Lombardo	
COSM	COSMOTE KINITES TILEPIKOINONIES AE
G. Lyberopoulos, E. Theodoropoulou, I. Mesogiti	
UBITECH	GIOUMPITEK MELETI SCHEDIASMOS YLOPOIISI KAI POLISI ERGON PLIROFORIKIS ETAIREIA PERIORISMENIS EFTHYNIS
P. Gouvas, A. Zafeiropoulos, T. Xirofotos	
INC	INCELLIGENT IDIOTIKI KEFALAIOUCHIKI ETAIREIA
N. Stasinopoulos, A. Ropodi, K. Tsagkaris, P. Demestichas	
ATOS	ATOS SPAIN SA
J. Melian Hernandez	
ERICSSON	ERICSSON TELECOMUNICAZIONI
O. Toscano	
EXXPERT	EXXPERT SYSTEMS GMBH
E. Beckmann, J. Albrecht, U. K. Adusumilli	
ORO	ORANGE ROMANIA
J. Ghenta, H. Stefanescu, M. Iordache	
ININ	INTERNET INSTITUTE Ltd
L. Koršič, D. Mulac, J. Cijan, J. Sterle	
ITL	ITALTEL
A. Albanese	
BIBA	BREMEN INSTITUT FUER PRODUKTION UND LOGISTIK GMBH
K. Burow, Z. Ghrairi	

Disclaimer

The information, documentation and figures available in this deliverable are written by the MATILDA Consortium partners under EC co-financing (project H2020-ICT-761898) 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

The MATILDA Consortium consists of:

CONSORZIO NAZIONALE INTERUNIVERSITARIO PER LE TELECOMUNICAZIONI (CNIT)

ATOS SPAIN SA (ATOS)

ERICSSON TELECOMUNICAZIONI (ERICSSON)

INTRASOFT INTERNATIONAL SA (INTRA)

COSMOTE KINITES TILEPIKOINONIES AE (COSM)

ORANGE ROMANIA SA (ORO)

EXXPERTSYSTEMS GMBH (EXXPERT)

GIOUMPITEK MELETI SCHEDIASMOS YLOPOIISI KAI POLISI ERGON PLIROFORIKIS ETAIREIA PERIORISMENIS EFTHYNIS (UBITECH)

INTERNET INSTITUTE, COMMUNICATIONS SOLUTIONS AND CONSULTING LTD (ININ)

INCELLIGENT IDIOTIKI KEFALAIOUCHIKI ETAIREIA (INC)

NATIONAL CENTER FOR SCIENTIFIC RESEARCH "DEMOKRITOS" (NCSRD)

UNIVERSITY OF BRISTOL (UNIVBRIS)

AALTO-KORKEAKOULUSAATIO (AALTO)

UNIVERSITY OF PIRAEUS RESEARCH CENTER (UPRC)

ITALTEL SPA (ITL)

BIBA - BREMER INSTITUT FUER PRODUKTION UND LOGISTIK GMBH (BIBA)

SUITE5 DATA INTELLIGENCE SOLUTIONS LIMITED (S5).

Table of Contents

Table of Acronyms

Acronym	Definition
API	Application Programming Interface
BER	Bit-Error Rate
CSM	Computing Slice Manager
eNB	Evolved Node-B
EPC	Evolved Packet Core
GUI	Graphical User Interface
IaaS	Infrastructure as a Service
ICMP	Internet Control Message Protocol
IOPS	Input/Output Operations per second
KPI	Key Performance Indicator
MEC	Mobile Edge Computing
NFV	Network Functions Virtualization
NFVCL	Network Functions Virtualization Convergence Layer
NFVO	Network Functions Virtualization Orchestrator
NS	Network Service
OSS	Operations Support Systems
PaaS	Platform as a Service
PHP BL	PHP: Hypertext Preprocessor Business Logic
PLMN	Public Land Mobile Network
PNF	Physical Network Function
PoP	*Point of Presence*
PPDR	*Public Protection and Disaster Relief*
RAN	Radio Access Network
RSO	Resource Selector Optimizer
SLA	Service Layer Agreement

Acronym	Definition
SW	Software
UC	Use Case
UE	User Equipment
UI	User Interface
VAO	Vertical Application Orchestrator
VIM	Virtual Infrastructure Manager
VNF	Virtual Network Function
WAN	Wide-Area Network
WIM	Wide-Area Infrastructure Manager

1 Executive Summary

The aim of MATILDA is to deliver "a holistic, innovative framework for design, development and orchestration of 5G-ready applications and network services over sliced programmable infrastructure". The MATILDA project aims to provide a solution as realisation of this framework by unifying network slicing, edge computing and multi-tenancy abstractions into an integrated system, by methodically following the lifecycle process of development, deployment and operation of 5G verticals' use cases. The entire vision will be demonstrated through a set of test cases chosen to highlight different verticals.

In the first version of this document, the MATILDA evaluation framework has been detailed as flows of validation and evaluation processes, spanning from **MATILDA Solution Components and Functionality Validation** to **General (as a whole) Solution Validation and Evaluation** and further to **Performance Evaluation on the basis of specific KPIs** of MATILDA specific functions and of the whole solution. Validation testing and evaluation flows are addressed at various completion degrees at various project stages, namely: at MATILDA component development phases, at MATILDA components' integration phases, at vertical application on-boarding phases, at MATILDA solution operational phases, at vertical application full deployment phases, and finally at vertical application operational phases. As planned, the list of test objectives and procedures are being refined throughout the project lifetime to better suit implementation specificities that emerge in these project stages, along with testbed specific features, environment setup/tools, etc.

This document leverages the previous version with the complete set of validation and performance evaluation results. While the first version of the document focused on the validation results of the majority of Solution Components and Separate Functionalities, this document provides validation tests of (1) the rest of the MATILDA components integrated solution testing, not only on CNIT/UBITECH, but also (2) on a number of MATILDA Demonstrators' testbeds, related to (3) Vertical Applications Deployment and Network Slice Lifecycle Management over a completely integrated MATILDA infrastructure.

Validation results retrieved are the following:

- The lifecycle management (insertion, modification/update, selection, deletion) of applications/application components/VNFs and their metadata in the associated repositories has been tested and successfully validated.
- The Vertical Applications' orchestration and lifecycle management has been tested (at UBITECH/CNIT testbeds), and successfully validated in terms of enabling Real-Time deployment of an application in various PoPs, including enforcement of specific run-time policies (resource utilisation and security-related).
- A number of Vertical Applications deployment monitoring functionalities have been tested (at UBITECH/CNIT testbeds and with a number of vertical Demonstrator applications) and the capability of monitoring compute/network resources utilisation and application behaviour from multiple sources and of extracting Analytics and advanced insights has been successfully validated.

- The lifecycle management of NSs is being finalised and tests have focused on 3GPP network services slice provisioning, and on MEC capabilities using a "Bypass VNF" enabling traffic offloading at edge PoPs; WAN–specific functionality related to the management of network resources on a per-slice basis is also under testing.

Complete demonstrations have been made with all MATILDA vertical applications, and significant hands-on experience has been acquired by the verticals as MATILDA end-users. As revealed from the experimentation, testing and evaluation results that have been collected from all demonstrators:

- The on-boarding, application deployment and lifecycle management processes of vertical applications have been considered as user-friendly and adequate in terms of network service definition and procedural steps for the partners/personnel involved in the project.
- The average deployment time has been significantly minimised, especially when considering the second / third /etc. time deployment, with first time installation and on-boarding the total time is 90min inline with the global 5G KPIs.
- QoS guarantees can be provisioned in terms of achieved maximum and guaranteed data rates, in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces; in line with 5G KPIs, as tested, data rates of 100Mbps for the MATILDA vertical applications can be guaranteed.
- QoS guarantees can be provisioned in terms of achieved latencies, in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces. Latencies ranging even between tens to 150ms, can be achieved depending on the network deployment and the network technologies materialising the slices in testbeds implementations.
- High number of connections can be supported, with the MATILDA project implementation achieving the support of at least 1000 IoT devices (physical or emulated).

This document provides the details related to testing procedures, KPIs, results on a per test case basis.

2 Introduction

The aim of MATILDA has been to deliver "a holistic, innovative framework for design, development and orchestration of 5G-ready applications and network services over sliced programmable infrastructure". To this end, the MATILDA project has provided a solution as realisation of this framework by unifying network slicing, edge computing and multi-tenancy abstractions into an integrated system by methodically following the lifecycle process of development, deployment and operation of 5G use case verticals. The MATILDA architecture comprises three distinct layers: the Development and Marketplace Environment, which supports all pre-deployment steps of a 5G-enabled application, including the vertical application development and wrapping and the application service graph creation, along with a set of runtime policies used during deployment; the Vertical Application Orchestrator (VAO), in charge of slice intent deployment delivery over the programmable infrastructure; and the Slicing and Management Programmable infrastructure, which is responsible for lifecycle management of the application graph deployment, using network and computing resources from the underlying infrastructure.

The entire vision has been demonstrated through a set of test cases chosen to highlight different verticals. The MATILDA evaluation framework has already been described in Deliverable D6.1 [MATILDA-D.6.1], where the different validation and evaluation phases (prior to demonstration) have been defined, along with the preliminary identification of a number of specific test objectives, the associated KPIs and the generic validation method to be followed, including the MATILDA components involved. According to the evaluation framework, tests spanned from component validation to functionality validation and evaluation and further to performance evaluation.

The test objectives defined in the framework have been addressed throughout the course of the project at various stages, namely: at MATILDA component development phases, at MATILDA components' integration phases, at vertical application on-boarding phases, at MATILDA solution operational phases, at vertical application full deployment phases, and finally at vertical application operational phases. As planned, this initial list of test objectives and procedures has been refined throughout the project lifetime to better suit implementation specificities that emerge in these project stages, along with testbed specific features, environment setup/tools, etc.

This deliverable concludes the experimentation and evaluation activities and provides an overview of the work done and results obtained in the context of Task 6.7. More specifically:

Section 3 provides an overview of the MATILDA Evaluation framework and a mapping between the objectives and the different project stages. This section puts special focus on the identification of the KPIs addressed in the context of the project evaluation activities.

Section 4 provides a refinement of the Solution Components and Functionality Validation-related objectives and specifies them at the level of tests, KPIs and success criteria, while it summarises the retrieved validation and evaluation results.

Section 5 provides a summary of the Solution Performance evaluation results, on the basis of vertical applications' on-boarding, deployment and lifecycle management at various demonstrator sites, performed in the context of Tasks Task 6.4- Task 6.8.

Section 6 provides the suitable references to the project work related to the provision of necessary guidelines addressing any tentative user of the MATILDA platform, to help him/her through the vertical applications' on-boarding, deployment and lifecycle processes.

Finally, **Section 7** provides a summary and conclusions of the document.

3 Evaluation Framework Overview

The initial MATILDA evaluation framework has been provided in [MATILDA-D6.1], and has been refined along with 1st phase evaluation results in [MATILDA-D6.7]. More specifically, the evaluation framework is structured around a number of test objectives, which reflect complete stakeholders' operational procedures (consisting of one or more MATILDA components/functionalities) or/and complete infrastructure operations (consisting of one or more MATILDA components/functionalities), as well as users'/stakeholders' performance and miscellaneous requirements to be satisfied. These test objectives are evaluated against their associated KPIs.

Complete testing associated with each test objective comprises a number of different validation and evaluation phases and testing procedures spanning from component to functionality validation and evaluation, and further to performance evaluation, specifically targeting:

- **Solution Components and Functionality Validation** aiming at verifying the operation and evaluating the performance of the functionalities/capabilities to be provided by a single (Component Functionality) or by multiple (Complex Functionality) components of the MATILDA solution ([MATILDA-D1.1] and [MATILDA-D4.2]).
- **General Solution Validation** aiming at the evaluation of the solution as a whole for the development and definition of 5G-ready (vertical) applications and NSs and their deployment over a sliced network infrastructure.
- **Performance Evaluation** of specific functions, as well as of the whole MATILDA solution, on the basis of specific applications' KPIs defined in the MATILDA use cases or/and by the MATILDA end-users, as well as towards the 5G-PPP KPIs [MATILDA-D1.1].

Therefore, the test objectives defined in the framework are addressed throughout the course of the project at various stages, in terms of being:

- refined throughout the project lifetime to better suit implementation specificities that emerge in the various stages,
- elaborated at specific test levels with specific success criteria, even
- tailored to specific testbed features, environment setup/tools, vertical application specificities, etc.

To define the workflow to realise the evaluation framework, we shall consider the MATILDA project implementation phases. In this respect:

- the **MATILDA component development** phases, in which "Solution Components and Functionality Validation" is performed;
- the **MATILDA components' integration** phases, in which "more complex Functionalities' Validation" and specific "Performance Evaluation" is performed;

- the **MATILDA solution operational phase**, in which "General Solution Validation" and "Performance Evaluation" is performed;
- the **vertical application development** (including transformation of application to become "5G-ready") phase, in which "Solution Components and Functionality Validation" and part of "Performance Evaluation" is performed;
- the **5G-ready (vertical) application on-boarding** phase, in which "Solution Components and Functionality Validation" and part of "Performance Evaluation" is performed;
- the **5G-ready (vertical) application deployment** phase, in which "Solution Components and Functionality Validation" and part of "Performance Evaluation" is performed;
- and, finally, the **5G-ready (vertical) application operational phase**, in which "Solution Components and Functionality Validation" and general "Performance Evaluation" is performed.

These project implementation phases, hence the related validation and evaluation activities, are not followed in a strictly sequential order. The following Figure 1 illustrates the generic approach and a mapping between the MATILDA implementation and the validation and evaluation activities.

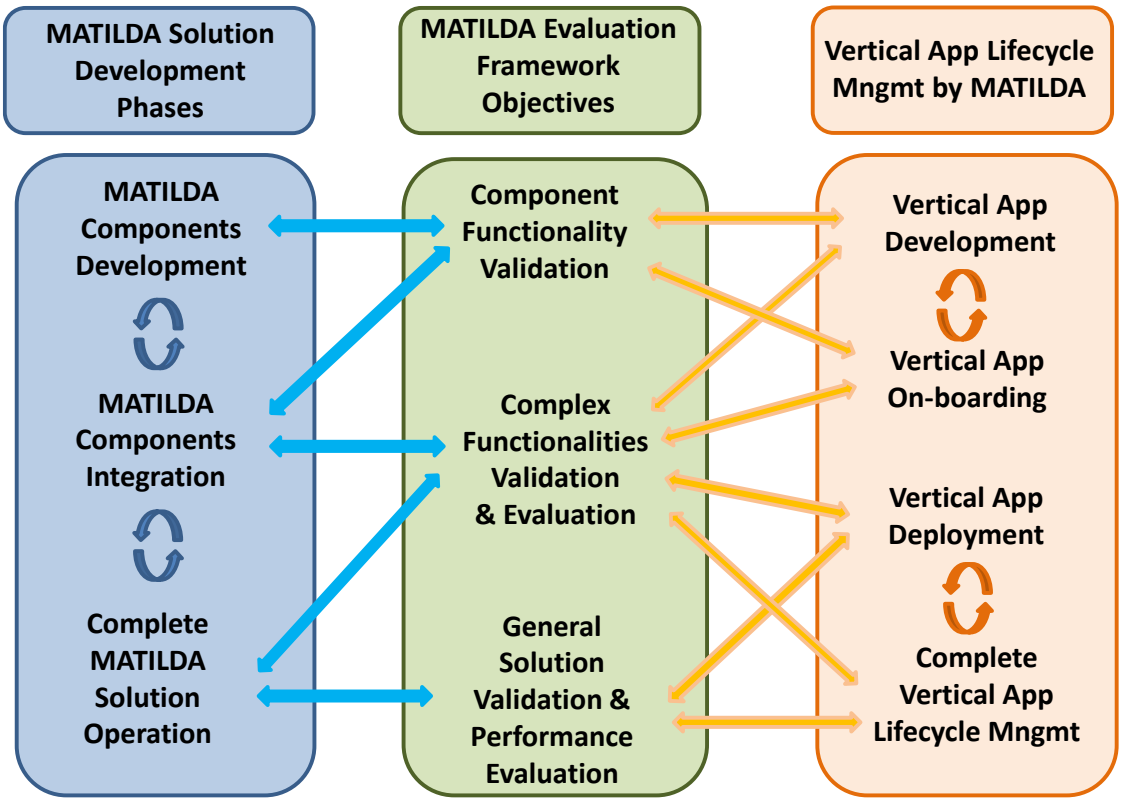


Figure 1: MATILDA Evaluation Framework Overview.

3.1 Elaboration of Test Objectives

The MATILDA test objectives have been initially defined in [MATILDA-D6.1], and further refined in [MATILDA-D6.7], and include the following complete operations and their performance at solution/ infrastructure level and at vertical application level:

- 5G-ready Applications Development.
- 5G-ready Applications Lifecycle Management in MATILDA repositories[1].
- Vertical Applications Orchestration including Deployment and Monitoring.
- Lifecycle Management of a Service Request.
- Lifecycle Management of Slices (including slice negotiation and orchestration of network and compute resources).
- Lifecycle Management of NSs (including VNFs).
- Management of Infrastructure Resources.
- Management of Wide-area Network Resources (including Multi-site Resource Management).

The MATILDA solution supports these operations through its main solution components as defined initially in [MATILDA-D1.1] and detailed in [MATILDA-D4.2]:

- *The 5G-ready applications development toolkit*, providing support for:
    a. the application/component development and wrapping,
    b. the various applications' service graphs' definition/creation/edition,
    c. the runtime policies creation/edition.
- *The MATILDA Marketplace/repositories*, providing the interface to end users/application owners/verticals and supporting:
    a. the lifecycle management of applications/application components' in the repository,
    b. the lifecycle management of VNFs in the repository,
    c. the handling of various, different profiles/operations for different users/ stakeholders/roles.
- *The Vertical Application Orchestrator (VAO)*, enabling:
    a. real-time vertical (5G-ready) application deployment planning; extraction of the slice intent on the basis of the MATILDA metamodels and negotiation of its properties taking under consideration the available programmable resources and the running infrastructure/resources status,

[1] In the context of this document, the term "vertical application" refers to an application owned/maintained by a vertical industry, and "5G-ready application" to an application that adheres to the MATILDA wrapping principles and metamodels. In some cases, these terms are used interchangeably, because in the context of the MATILDA project all applications used/tested/demonstrated are transformed to 5G-ready version, while representing/belonging to a vertical industry/partner.

b. enforcement of specific execution policies over the deployed vertical application following a continuous match-resolve-act approach,
c. monitoring and management of applications/application components through Monitoring and Data fusion mechanisms, and
d. extraction of advanced insights and events from the analytics data of the Monitoring process, for support of re-active reconfigurations (manually or automatically) of application deployment,
e. the lifecycle management of the applications (application components) deployment.

- The *Operations Support System (OSS)*, in charge of:
  a. receiving the slice intents from vertical applications (i.e. from the VAO)
  b. coordinating the work of all the other building blocks in the Telecom layer platform to set up and to properly configure base 4/5G network services, network slices, and edge computing resources.

- *The NFV Orchestrator (NFVO)*, in charge of:
  a. managing the lifecycle of the network services composing the base 4/5G services, and of the ones provided to slices in a shared or isolated fashion,
  b. Day-2 operations for PNFs (e.g., g/eNodeBs).

- The *Virtual Infrastructure Manager* (VIM - one instance per each distributed computing facility) devoted to abstract and expose computing, storage, and networking capabilities of datacenters within the 5G infrastructures. It has the key role of isolating the various tenant domains (i.e., NFV domains and Vertical Applications' ones), as well as of creating shared resources to properly "attach" these domains.
- The *Wide-area Infrastructure Manager* (WIM), devoted:
  a. to manage and monitor the wide-area communication resources (through southbound interface to networking devices either directly or through SDN controllers),
  b. to create network overlays to be used in a shared or isolated fashion by vertical applications and base telecommunication services, as well as
  c. to provide information on which resources (e.g., VIMs, PNFs, etc.) can be selected in the distributed 5G infrastructure to create slices/services in order to satisfy vertical application performance requirements (e.g., end-to-end latency, bandwidth, etc.).

- The *Wide-Area SDN Controller* (WSC), in charge of interconnecting the control agents of the SDN devices in the wide-area network for monitoring and configuration purposes.

With the finalization of the MATILDA components' development and their integration, the Solution Components and Functionality Validation-related objectives have been refined and apart from being elaborated at the level of specific component functionality and complex functionalities validation tests' and success criteria, they have been linked to specific target 5G-network KPIs aligned with the global and EU efforts in this area. The functionality testing and results, along with the KPIs evaluation are included in this deliverable (esp. Chapter 4).

The test objectives related to the users'/stakeholders' performance and miscellaneous requirements have been also defined initially in [MATILDA-D6.1], and have been refined at

the level of specific tests' and success criteria and evaluated throughout the course of the project through end-users' interaction with the solution at various stages namely:

- the transformation of the MATILDA vertical applications into 5G-ready applications,
- the on-boarding process of these applications' graphs,
- the deployment of the service graph,
- the service operation,
- the service (service graph) termination.

End-to-end performance evaluation, user friendliness, speed of application deployment, expandability of the solution, scalability, reliability, and so on are the objectives associated to the General Solution testing performed on a per demonstrator basis. The testing procedures, the results and the evaluation are detailed in deliverables [MATILDA-D6.8] to [MATILDA-D6.12].

3.2 Evaluation of Results and KPIs

3.2.1 Overview of 5G Network and Services KPIs

In line with the MATILDA evaluation framework as afore described, and throughout the course of the project iterative testing, validation and evaluation phases, the solution evaluation has been based on the proper operation validation of MATILDA functionalities, as well as on the evaluation of their performance on the basis of specific, measurable, understandable KPIs. To this end, the MATILDA project has closely monitored the work performed by Standardisation Organisations (SDOs) mainly 3GPP, ETSI and ITU, as well as industry alliances and fora. In particular, ITU initially provided the applications and network services generic target KPIs as illustrated in Figure 2.

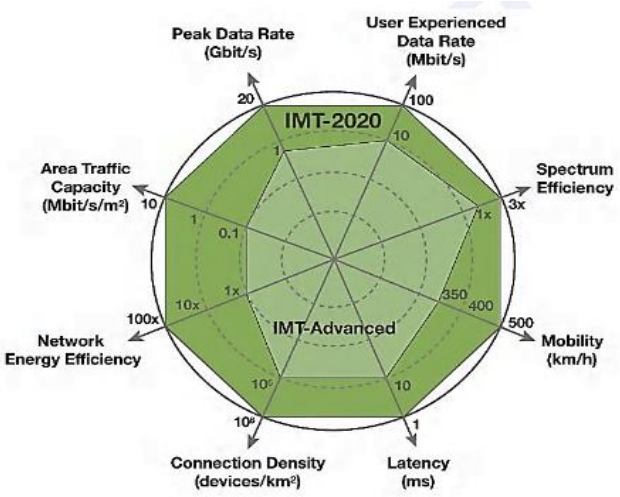


Figure 2: IMT 2020 Targets by ITU [ITU-R M.2083]

Refining further these generic targets, ITU and 3GPP distinguished generic Service Classes to be used for mapping vertical applications/services (also identified by 3GPP/NGMN [NGMN 5G]/etc.), namely: enhanced Mobile BroadBand (eMBB), massive/enhanced Machine Type

Communications (m/eMTC) or massive Internet of Things (mIoT) and Ultra Reliability Low Latency Communications (URLLC).

While adhering to these classes, 3GPP defined various KPIs on a per service/vertical category/ deployment/ etc. basis along with target values in [TS 22.261], [TR 22.861], [TR 22.862], [TR 22.863], [TR 22.864], [TR 22.891], with the purpose to be widely used for the assessment of 5G network capabilities and deployments.

In particular, the mIoT Service Class is associated with massive IoT deployments e.g. for monitoring, surveillance, alerting sensors, industry 4.0 as well as for automotive vertical use cases [TR 22.861]. Obviously, different IoT deployments and applications have different data rate, latency and connection density requirements thus target values for these KPIs. URLLC Service Class reflects the requirements of mission critical communications e.g. in industry automation, medical application, etc., requiring high reliability and availability as well as low latency (e.g. <1ms in control plane and <10 ms in user plane) [TR 22.862]. Finally, the eMBB Service Class is reflecting the requirements of high data rate services (multimedia services, broadband data services) reaching 100Mbit/s per user data rate, and areas/hotspots of high traffic density reaching 3.75 Mbit/s DL/m^2 and 7.5 Mbit/s/m^2 UL [TR 22.863].

At the same time, in the context of early activities of 5G-PPP, the infrastructure and services' KPIs defined by standardization bodies and industry alliances have been compiled, resulting in high-level, operational 5G network deployment KPIs [5G Broch], taking into account both 5G equipment capabilities and network deployment specificities. With the progress of work on 5G in various bodies and projects, all these KPIs have been overloaded with definitions and target values directly linked to specific applications (rather than verticals) and testing facilities (rather than deployments). Therefore, later stages work of 5GPPP focused on and continue working on the refinement of ITU and 3GPP KPIs, and their decomposition to specific, measurable, and relevant KPIs. Initially Latency and Service Deployment Time have been refined [5G PPP PMR]. In particular:

- The End to end latency KPI has been decomposed into a number of delays' (latencies) corresponding to network segments' transmission, network components' (equipment) processing and application layer processing.
- The Service Deployment Time KPI has been decomposed into timing KPIs related to the following distinctive operations: Platform Provisioning, Service/Application Onboarding, Service Instantiation/Configuration and Activation, Service Modification and Service Termination.

MATILDA has monitored and contributed significantly to the work performed in the context of 5G-PPP TB regarding the 5G KPIs. In particular, this work included:

1. Early identification, definition, and iterative refinement of the KPIs to be measured and evaluated in 5G infrastructures, along with information regarding the measurement points and methods. Major contribution to this work has been provided by MATILDA and initially published in "K.X. Du, B. Sayadi, G. Carrozzo, F. Lazarakis, A. Kourtis, M.S. Siddiqui, J. Sterle, O. Carrasco, and R. Bruschi, "Definition and Evaluation of Latency in 5G:A Framework Approach", URL: http://www.jkjmanagement.com/5gwf19-4/papers/p135-du.pdf, 2019 IEEE 5G World Forum", and secondly in "5G PPP phase II KPIs – Annex to Programme Management Report".

2. Alignment of this work with MATILDA project testing and evaluation activities, especially mapping it to specific demonstrators, work-packages, and demonstration activities; and communication of this information to the TB KPIs WG. This information has been reported in PPP Programme Management Report (finalised June 2019) (i.e. 5G-PPP Phase 2 KPIs – Annex to Programme Management Report).

In the opposite direction, the work performed and information obtained in the context of TB KPIs WG, has been fused in the MATILDA demonstrators' and solution validation and evaluation activities.

3.2.2 Evaluation of Vertical Application Lifecycle Management & KPIs in MATILDA

In parallel with the iterative testing, validation and evaluation phases of the MATILDA components and solution, a number of vertical applications have been prepared -in terms of being evolved from originally stand-alone applications to cloud-native ones- to be used for the MATILDA solution performance evaluation from the verticals perspective. The selected verticals and applications span on various service classes and application categories -in view to the aforementioned classification. To this end, initial service KPIs and target values have been identified and initially reported in [MATILDA-D1.6]. Latter KPIs have been further refined from the initially specified definitions and/or target values (summarised in [MATILDA-D6.7]), due to the following reasons:

- At initial stages, KPIs were defined considering the requirements of general applications falling in the category of each demonstrator; as the project progressed, the applications to be finally used were further elaborated, thus KPIs (especially network ones) were further nailed down to the performance requirements of the specific applications to be tested.
- The initial KPIs referred to traditional applications instantiation (e.g. in a single server, single client mode), while in the context of the project the applications were also modified to include a number of cloud-native components; thus, in many cases, it was required to move from single-link KPIs to application graph KPIs (also reflected in the metamodels).
- Throughout the course of the project, even the expertise of partners has advanced to obtain a better understanding of the requirements and KPIs that they shall expect to be achieved by the solution. To this end, even new KPIs, especially related to the operation of the MATILDA solution (below defined as operational KPIs) and the rapid deployment and instantiation of an application, have been considered and added in the initial list - mainly where it has been considered a critical parameter for the demonstrator. Indicatively, the deployment time (operational KPI) has been added in the case of the 5G PPDR application that is usually deployed on demand, contrary to the smart lightning application that does not require rapid on demand deployment.
- On the other hand, in certain cases, specific KPIs were defined, which however cannot be demonstrated by the project test facilities and/or are out of scope of the core of the MATILDA solution, but rather need to be evaluated in larger and completely operational environments. These KPIs were removed or their values have been changed to be more applicable to the specific, available test facilities, features, environment setup/tools, etc.

The KPIs on a per demonstrator phase and the overall evaluation is elaborated in Chapter 4 of this document. Later stages of the MATILDA evaluation framework focused on the

assessment of the performance of the vertical services in view of the aforementioned KPIs, when they are deployed over a MATILDA orchestrated infrastructure and their lifecycle is managed by MATILDA. The KPIs on a per demonstrator phase and the overall evaluation is elaborated in Chapter 0 of this document.

3.2.3 Evaluation of KPIs in MATILDA

Considering the 5G networks and network services KPIs defined by ITU, NGMN, 3GPP and 5GPPP, MATILDA solution evaluation has addressed the following:

Table 1: 5G KPIs Evaluation in MATILDA.

KPIs		Source	MATILDA Focus	Comments
Multitenancy support		3GPP & 5GPPP	√	Demos
Co-existence with heterogeneous processing capabilities		MATILDA	√	Demos with: (1) CPUs, GPUs, (e.g. 5G-PACE app); (2) various 3GPP & non-3GPP networks (e.g. PPDR app)
Scalability		5G-PPP / Telcos	√	
Number of connected devices		ITU, 3GPP	√	ORO Demo of IoT UC
Radio Network node capacity		5G-PPP		Out of scope – the project focused on networks' compute slicing and OSS. Due to the project timing, LTE/WiFi access network nodes were used
Latency	**T_RAN Latency**	5G-PPP		
	T_Backhaul	5G-PPP		
	T_Core	5G-PPP		
	P_UE	5G-PPP		
	P_RAN	5G-PPP		
	P_UPF_Edge	5G-PPP		
	P_UPF_Core	5G-PPP		
	R_Client	5G-PPP		
	R_Server_Edge	5G-PPP		
	R_Server_Core	5G-PPP		
User data rate		ITU, 3GPP	√	
Reliability		ITU, 3GPP	-	Out of scope. Such KPIs are assessed at production network deployments
Availability		ITU, 3GPP	-	

KPIs		Source	MATILDA Focus	Comments
Ubiquitous access		5G-PPP	-	
Mobility		3GPP, 5G-PPP	-	Out of project focus
Positioning Accuracy		ITU, 3GPP	-	Out of project scope.
Service Deployment Time				
Phase 0. Platform Provision	**Platform configuration**	5G-PPP	√	Configuration of Access network node (as platform) in PPDR Demo
	Platform deployment	5G-PPP	√	Deployment of Access network node (as platform) in PPDR Demo
Phase 1. Onboard-ing	**Network Slice Template**	5G-PPP	√	Definition of slice in OSS
	Network Service Descriptor	5G-PPP		
	VNF package	5G-PPP	√	Onboarding of VNFs in NFVO
	MEC App Descriptor	5G-PPP		
	Other applications	5G-PPP	√	Onboarding of VNFs in VAO
Phase 2. Instantiate Configure & Activate	**Instantiate Network Slice (NSI)**	5G-PPP	√	
	Instantiate & Activate Network Service (NS)	5G-PPP	√	
	Instantiate & Configure VNFs in service chain (VNF)	5G-PPP	√	
	Instantiate & Configure MEC App	5G-PPP	√	
	Instantiate & Configure other applications	5G-PPP	√	
	Configure other NFVI elements	5G-PPP		
	Configure SDN infrastructure	5G-PPP	√	
	Configure Optical WAN	5G-PPP	-	Out of project scope
	Configure satellite backhaul	5G-PPP	-	Out of project scope

KPIs		Source	MATILDA Focus	Comments
Phase 3. Modify	Modify Network Slice configuration	5G-PPP	?	
	Modify Network Service configuration	5G-PPP	?	
	Detect scale out/in decision	5G-PPP	√	
	Implement manual scale out/in	5G-PPP	√	
	Implement autoscale out/in	5G-PPP	√	
	Modify VNF configuration in service chain	5G-PPP		
	Modify MEC App configuration	5G-PPP		
	Modify configuration of other applications	5G-PPP		
	Modify configuration of other NFVI elements	5G-PPP	-	Out of project scope
	Modify configuration of SDN infrastructure	5G-PPP	√	
	Modify Optical WAN circuit	5G-PPP	-	Out of project scope
	Modify satellite backhaul configuration	5G-PPP	-	Out of project scope
Phase 4. Terminate	Terminate Network Slice	5G-PPP	√	
	Terminate Network Service	5G-PPP	√	
	Terminate VNFs in service chain	5G-PPP	√	
	Terminate MEC App	5G-PPP	√	
	Terminate other applications	5G-PPP	√	
	Remove configuration of other NFVI elements	5G-PPP	-	Out of project scope

KPIs		Source	MATILDA Focus	Comments
	Remove configuration from SDN	5G-PPP	-	Out of project scope
	Terminate Optical WAN circuit	5G-PPP	-	Out of project scope
	Terminate satellite backhaul circuit	5G-PPP	-	Out of project scope

3.3 Revised Tests Definition Templates

For the purpose of having a homogeneous description of the test objectives, tests and results to be performed in the context of MATILDA have been specified by the contents of the following (Table 2) fixed format tables (the first one also defined in [MATILDA-D6.1], but repeated in this section to facilitate the reading of the rest of the document).

Table 2: Tests Definition in Tabular Format.

Test Objective	<#>	**Type**	<End User Performance/ Functionality / Solution Components / General Solution>
Title	<Title of the Tests.>		
Relevant UCs	<UC #> (applicable only for the End-User Performance tests)		
Validation method – Tests	<Description of the validation method and definition of tests.>		
KPIs	<KPIs and success criteria.>		
Components	<MATILDA solution components; where applicable.>		
Test bed	<Test bed to perform the tests> (if known at this stage)		

Detailed Tests Description: <Title of Test Objective>	
Test # <Component Testing> OR <Component Testing & Functionality Evaluation>	**Description:** <Elaboration on the tests to be performed towards validating or/and evaluating the associated test objective (described in the previous table).> **Success Criteria:** <Definition of success criteria for the tests of type <Component Testing>, against which results will be validated.> **Evaluation KPI:** <Definition of KPIs for the tests of type <Functionality Evaluation>.> **Testbed:** <Test bed to perform the tests.>
Results/ Comments	<Description of the expected or obtained results from the associated test, and other relevant comments.>

The following information is associated with the fields of the tables:

- **Number:** This field provides an increasing number to exclusively identify each individual test/set of tests with a specific scope, to ease tracking of its fulfilment in the next steps of the project.
- **Type:** Indicates the category of the test.
- **Title:** The title of the test practically corresponds to the testing purpose.
- **Relevant UCs:** Identifies the Use Case (UC) to which this test is related, and is applicable only to the end-user performance tests.
- **Validation method – Tests:** Provides a brief description of the validation method to be followed and the tests to be performed.
- **KPIs:** Defines the KPIs and the criteria or/and values to evaluate the success of the tests.
- **Components:** Defines the components of the MATILDA solution that are involved or which will be tested. This field is mainly applicable to MATILDA solution and functionality testing.
- **Test bed:** Defines the test bed in which these tests will be performed.

A second table is associated with each test objective (table), which includes the list of elaborated tests (**Test #** - field) to be performed towards validating or/and evaluating the objective against specific success criteria. In this document version this table format has been revised in order to provide a clear distinction between the <component validation> and <functionality evaluation> tests; the latter being the focus also of the 2nd phase testing activities. Specific Evaluation Functionality KPIs are also defined in this table. This table is also used to collect the obtained results of the tests (**Results/Comments** - field).

4 MATILDA Component-level Validation and Functionality-level Validation and Evaluation

This section provides a refinement of the Solution Components and Functionality Validation-related objectives, as identified and numbered in [MATILDA-D6.1], and an elaboration of these objectives at the level of specific tests' and success criteria. Given the fact that, currently, the development of a number of MATILDA components has been finalised, and some of them have been partially integrated, preliminary test results have been obtained. The latter are summarised in the following tables and figures.

Table 3: 5G-ready Applications Development.

Test Objective	5	**Type**	Functional
Title	5G-ready applications development using MATILDA Toolkit		
Validation method – Tests	Tests related to the 5G-ready applications' development, in particular to: • application component development and wrapping to transform it to cloud-native, including code/wrapping verification, and assessment of the MATILDA Development and Wrapping Toolkit; • creation/edition of application service graphs adhering to the MATILDA metamodels to transform it to 5G-ready, including verification of understandability, completeness, assessment of metamodels' and MATILDA Application graph editor; • creation/edition of runtime policies at application component level, through the MATILDA Policy Editor.		
KPIs	**Success Criteria:** Successful migration of an on-premises developed application to a 5G-ready version by using the MATILDA 5G-ready Application Development Toolkit.		
Components	Development and Wrapping Toolkit, Application Graph Editor, Policy Editor.		
Testbed	UBITECH, CNIT		

Detailed Tests Description: 5G-ready applications development using the MATILDA Toolkit	
Test 1 Component Testing	**Description:** Design and development of components of a 5G-ready application by using the MATILDA toolkit. Application/component development and wrapping so that it becomes cloud-native, including code/wrapping verification, and assessment of the MATILDA Development and Wrapping Toolkit. **Success Criteria:** Error-free wrapping of application components' SW code. Availability of the developed application components in the associated MATILDA components' repository. **Testbed:** UBITECH – verification (1) at MATILDA Development and Wrapping Toolkit at development and (2) at MATILDA Demonstrators' applications development phases.

Result/ Comments	Tests have been performed during the first phase testing, focusing on the design, development and registration in the MATILDA Repositories of the components of the MATILDA demonstrators' applications as reported in [MATILDA-D6.2]-[MATILDA-D6.7].
Test 2 Component Testing	**Description:** Creation/edition of application service graphs adhering to the MATILDA metamodels. Verification/validation of the Application Graph Editor functionality, in terms of: • Capability to select a number of application components from the MATIDLA components' repository to form the application; • Capability to define the links/communication interfaces between them; • Capability to define/edit the application components' characteristics and requirements (incl. execution requirements, interfaces, etc.) as defined in the Chainable Application Component & 5G-ready Application graph metamodels; • Capability to define all the network resource requirements as defined in the Network-aware Application Graph Metamodel. **Success Criteria**: Support of the aforementioned functionalities/capabilities. **Testbed**: UBITECH.
Result/ Comments	Tests have been performed with the first version of the Graph Composer during the first phase testing as reported in [MATILDA-D6.2]-[MATILDA-D6.7].
Test 3 Component Testing	**Description:** Creation/Editing of application/components' elasticity runtime policies' definition during the design time through the MATILDA Policy Editor. **Success Criteria**: Capability to specify valid runtime policies based on the usage of the Policy Editor. **Testbed**: UBITECH/CNIT/Demonstrator testbeds.
Result/ Comments	As defined in [MATILDA-D1.5], a set of policies should be specified using the Policy Editor, and the expressions are validated immediately upon saving. This functionality was tested for a number of policies triggering scaling-out actions on the basis of compute resources utilisation thresholds, using the PPDR Demonstrator Application as reported in [MATILDA-D6.7].
Test 4 Component Testing	**Description:** Creation/Editing of application/components' security runtime policies definition during the design time through the MATILDA Policy Editor. **Success Criteria**: Capability to define the aforementioned runtime policies based on security criteria. **Testbed**: UBITECH.
Result/ Comments	Tests have been performed with the MATILDA Policy Editor with the security policy defined to "alert host on ICMP package arrival", as reported in [MATILDA-D6.7].

Test 5 Component Testing	**Description:** Extraction of Slice attributes from the MATILDA application components' profiling functionality. **Success Criteria**: Capability to define the aforementioned runtime policies based on application components' profiling. **Testbed**: UBITECH.
Result/ Comments	Tests have been performed at MATILDA profiling functionality development phase using internal applications as reported in [MATILDA-D6.7].

Table 4: 5G-Ready Applications' Lifecycle Management in the MATILDA Marketplace.

Test Objective	6	**Type**	Functional
Title	5G-Ready Applications' Lifecycle Management in the MATILDA Marketplace		

Validation method – Tests	Tests to be performed are related to the lifecycle management of 5G-ready applications/ components/ VNFs through the MATILDA Marketplace, including:
	• verification of the interface to end-users/application owners/verticals in terms of including all necessary functionality for these stakeholders
	• the lifecycle management of applications/application components/VNFs modules and their metadata in the repository, including:
	  • insertion,
	  • modification/update,
	  • selection,
	  • deletion,
	  • users' access rights definition/alteration.
	• the handling of the user rights for various, different profiles/functions for different users/stakeholders/roles.		

KPIs	**Success Criteria:** Successful performance of the functionalities that have been specified to be performed through the MATILDA marketplace interface on a per user/role basis. Consistency maintained between the information shown through GUIs with the actual repository information, and the specified rules on a per user/role basis.		
Components	MATILDA Marketplace interface, Component Repository, VNF repository, Application Graph Repository.		
Testbed	All.		

Detailed Tests Description: 5G-Ready Applications' Lifecycle Management Testing	
Test 1 Component Testing	**Description:** Registration of Infrastructure Resources/Domains on which a 5G-ready application can be deployed. **Success Criteria**: Capability to register different types of resources/domains to enable the design/deployment of components/graphs. **Testbed**: UBITECH/CNIT/Demonstrator testbeds.

Result/ Comments	A set of resources are registered and made available for deployment as reported in [MATILDA-D6.7]. The registration of resources has been tested under various infrastructure types like: Amazon Web Services, Google Cloud and OpenStack, and testbeds CNIT/UBITECH.
Test 2 Component Testing & Functionality Evaluation	**Description:** Verification/validation of performing 5G-ready applications' Lifecycle Management procedures through the MATILDA Marketplace, including: • insertion of a new application/ application component, • modification/update, • selection/query, • deletion. **Success Criteria**: Capability to perform the aforementioned 5G-ready applications' Lifecycle Management procedures through the MATILDA Marketplace. **Testbed**: UBITECH/ATOS/CNIT.
Result/ Comments	Insertion, editing, modification and removal of application components, applications and graphs in the associated MATILDA repositories was heavily tested and the functionalities were successfully verified throughout the MATILDA Demonstrators' applications' on-boarding phases; detailed descriptions can be found in [MATILDA-D6.2]-[MATILDA-D6.6].
Test 3 Component Testing & Functionality Evaluation	**Description:** VNFs Lifecycle Management in the MATILDA Marketplace, including: • insertion of a new VNF/NS, • modification/update, • selection/query, • deletion. **Success Criteria**: Capability to perform the aforementioned VNFs' Lifecycle Management procedures in the MATILDA Marketplace. **Evaluation KPI:** **5G-PPP Phase 1. Onboarding >> VNF Package** • Speed of VNF Package Onboarding, • Speed of VNF Onboarding. **Testbed**: UBITECH/ATOS/CNIT.
Result/ Comments	Insertion, editing, modification and removal of VNFs in the associated MATILDA repository was initially tested at ATOS testbed and the functionalities were successfully verified. For the onboarding speed, please refer to Table 8, Tests 1 and 2: the onboarding process concludes Day-0 operations and take less than a second (839 ms for the PLMN, 325 ms for the Vyos). The onboarding of the single VNF is not detected by OSM but it represents a fraction of these figures.

Test 4 Component Testing	**Description:** Handling of user rights for various, different profiles/functions for different users/stakeholders/roles: • testing of VAO access from different end-users/verticals; • testing of NFVO access from specific users as network service providers; • testing of Applications/Application components from different end-users/verticals; • testing of VNFs access from different end-users/verticals. **Success Criteria**: Capability to perform the aforementioned secure access to various MATILDA components & Application & VNFs repository objects. **Testbed**: UBITECH/ATOS/CNIT/Demonstrators' testbeds.
Result/ Comments	Insertion, editing, modification and removal of VNFs in the associated MATILDA repository was initially tested at ATOS testbed and the functionalities were successfully verified. Further tests have been performed during the 2nd phase, with the complete integration of the MATILDA Marketplace VNFs repository with the WIM and VIMs, as demonstrated during the Ljubljana and Bucharest demos.

Table 5: Vertical Applications' Orchestration and Lifecycle Management.

Test Objective	7 & 9 (unified)	**Type**	Functional
Title	Vertical Applications' Orchestration		
Validation method – Tests	Tests to be performed are related to the real-time deployment of a 5G-Ready, Vertical Application through MATILDA, including: • the extraction of the slice intent from the service graphs definitions on the basis of the MATILDA metamodels; • delivery of Real-Time deployment planning of the vertical application components optimized by taking into account: the application service graph, the relevant execution policies, the programmable resources availability in various PoPs and the network resources' availability; • enforcement of specific execution policies over the deployed vertical application following a continuous match-resolve-act approach, based on monitoring data and analytics; • termination of application instance operation upon request.		
KPIs	**Success Criteria:** Successful performance of the functionalities related to the optimized, real-time deployment of a (5G-ready) Vertical Application, in terms of requested resources and provisioned ones taking into account the infrastructure capabilities. Successful performance of the functionalities related to the re-active reconfiguration of a (5G-ready) Vertical Application deployment. Successful performance of the functionalities related to Vertical Application termination.		
Components	VAO, Optimisation Engine, Policy Engine, Intelligent Proxy, Execution Manager		
Testbed	All.		

Detailed Tests Description: Vertical Applications' Orchestration Testing	
Test 1 Component Testing & Functionality Evaluation	Lifecycle management of a Vertical Application, including: • instantiation of Application Graph, • modification (if needed) of Application Graph, • termination/deletion of Application Graph. **Success Criteria**: Capability to apply and monitor lifecycle management functions during the overall lifecycle of a Vertical Application. **Testbed**: UBITECH/Demonstrator testbeds.
Result/ Comments	The Vertical Application lifecycle management includes operations that span from the application graph instantiation, where each application component is initially loaded, up to the termination of the application provisioning, where all application components are terminated/deleted. As reported also in [MATILDA-D6.7], in these tests, it was verified that upon request the VAO can spawn successfully a VM per component and that the component's dependencies are fulfilled. Afterwards, each component enters an operational phase until the termination of application provisioning is decided.
Test 2 Component Testing	**Description**: Verification that the slice provisioning (resources (slice) that are initially allocated) to an application/application graph upon its deployment/instantiation on infrastructure is in accordance with the slice intent. **Success Criteria**: Correct deployment of application components, so that the application is functional upon instantiation triggered by the VAO. Consistency maintained between the VAO information and the actual application/application graph state. **Testbed**: UBITECH/Demonstrator testbeds.
Result/ Comments	Proper slice provisioning has been thoroughly tested. It was verified, that the VAO is able to launch the VMs composing the application with the correct resources as defined in the Slice Intent, and that they communicate properly among each other.
Test 3 Component Testing & Functionality Evaluation	**Description:** Enforcement of elasticity (runtime) policies at component/application level, defined through the MATILDA Policy Editor and verification of the deriving actions' triggering, in particular for the following runtime policies: • Scale out in case of exceeding resources utilisation: definition of threshold and margin (time - resources) for various parameters (e.g.), verification of scale out performance. **Success Criteria**: Capability to enforce the aforementioned runtime policies based on resource criteria. **Testbed**: UBITECH/Demonstrator testbeds.
Result/ Comments	The actual enforcement of the specified policies was tested over a running instance of the PPDR application and the triggering and enforcement of the specified rules and actions was successful as reported in [MATILDA-D6.7].

Test 4 Component Testing	**Description:** Enforcement of security (runtime) policies at component/application level, defined through the MATILDA Policy Editor, and verification of the deriving actions' triggering, in particular for runtime policies based on **detection of alert or intrusion mechanisms.** **Success Criteria**: Capability to enforce the aforementioned runtime policies based on security related criteria. **Testbed**: UBITECH.
Result/ Comments	The actual enforcement of a security policy -defined to "alert host on ICMP package arrival"- was tested over a running instance of an application and the triggering and enforcement of the specified rules and actions was successful as reported in [MATILDA-D6.7].
Test 5 Component Testing & Functionality Evaluation	**Description:** Verification of proper termination of application instance operation including the update of VAO with regard to the application status and the release of allocated resources. **Success Criteria:** Upon trigger (e.g., manually from VAO), correct termination/deletion of the application instance and release of the allocated resources. Consistency maintained between the VAO information and the actual application/application graph state. **Testbed**: UBITECH –verification (1) at MATILDA VAO development phase, (2) at PPDR Demonstrator application deployment phase and at the next stage (3) at other MATILDA demonstrators' application deployment phases.
Result/ Comments	Tests have been performed to verify the correct termination/deletion of the application instance and release of the allocated resources, triggered manually from the VAO, and it was successfully verified that resources were correctly released and consistency was maintained between the information presented in VAO and the actual status of the application.

Table 6: Vertical Applications' Deployment Monitoring.

Test Objective	8	**Type**	Functional
Title	Vertical Applications' Deployment Monitoring		
Validation method – Tests	Tests to be performed are related to the real-time and historical monitoring of a Vertical Application deployment, including: • real-time monitoring of multiple applications/application components through a set of active and passive probes; • incorporation of monitoring processes defined in the application service graphs/metamodels; • Fusion of monitoring data coming from multiple parallel data loads from multiple sources; • support of Real-Time Analytics of multiple contexts; • extraction of advanced insights and events from the monitoring process, e.g. through data mining, as well as predictive and prescriptive analytics		

	mechanisms (i.e. regression, clustering or classification algorithms); • evaluation of the extracted information in terms of validity, usefulness, versatility, effectiveness and sophisticated processing.
KPIs	**Success Criteria:** Successful performance of real-time monitoring of multiple applications/ application components • Extraction of Real-time Descriptive Analytics for all the application components; • Data Fusion of data coming from all application components; • Generation of Real-time Predictive Analytics for metrics coming from all application components; • Representation of Fused Descriptive and Predictive Analytics on an Analytics Dashboard to support infrastructure DevOps and development decision making. The obtained results/information are valid, useful, versatile depending on the nature of the application, effective towards undertaking corrective actions, and processing is sophisticated leading to advanced conclusions.
Components	VAO, Stream Aggregator, Data Fusion and Real-time Analytics.
Testbed	All.

Detailed Tests Description: Vertical Applications' Deployment Monitoring Testing	
Test 1 Component Testing	**Description:** Verification of incorporation of monitoring processes defined in the application service graphs/metamodels. Such monitoring processes can be included either as application components or a VNFs and it shall be suitable to be configured accordingly. **Success Criteria:** Verification of definition/development of monitoring processes to be included either as application components or as VNFs. **Testbed:** UBITECH/Demonstrators' testbeds.
Result/ Comments	It was verified that the netdata plugins denoted in the application components specification (at the application components' wrapping phase) are activated and are providing relevant monitoring data to Prometheus as reported in [MATILDA-6.7].
Test 2 Component Testing	**Description:** Verification of real-time monitoring of multiple applications/application components through a set of active and passive probes. 1. Testing of proper initiation of active and passive probes. 2. Testing of configuration of active and passive probes (e.g. in terms of measurements' interval per parameter, measurement window, interface to retrieve measurement, etc.) 3. Testing of retrieval of the following parameters: • Compute Resources utilisation in terms of: CPU, RAM, IOPS, etc. • Network Resources utilisation in terms of: bandwidth, latency, etc. • Application/Application components load in terms of: function calls, APIs' utilization, open connections, database query load, application latency, etc.

	Success Criteria: Correct retrieval of measurements of the aforementioned parameters (correctness in terms of data and in terms of being in accordance with the probes' configuration). **Testbed:** UBITECH/Demonstrators' testbeds.
Result/ Comments	Monitoring is supported based on a set of netdata plugins, while monitoring data is collected and provided by Prometheus as reported in [MATILDA-D6.7] indicatively for the PPDR applications.
Test 3 Component Testing & Functionality Evaluation	**Description:** Verification of fusion of monitoring data coming from multiple parallel data loads from multiple sources, including: 1. Testing of validity of data and retrieval of a set of listed performance/utilisation/etc. parameters (KPIs) for all application components by the pub-sub mechanism; 2. Testing of correlation between performance/ utilisation/etc. parameters (KPIs) in each component and across components; 3. Testing the validity of composite/aggregated performance/ utilisation/etc. parameters (KPIs) that feed next stages of descriptive and predictive real-time analysis; 4. Testing of the scalability of the fusion and retrieval procedures, as well as the historical persistence of aggregated metrics. **Success Criteria:** Correct fusion of monitoring data coming from multiple parallel data loads from multiple sources. **Testbed:** UBITECH/Demonstrators' testbeds.
Result/ Comments	In the 1st testing phase the aforementioned tests were performed with the PPDR application and obtained results have been reported in [MATILDA-D6.7]. During the 2nd testing phase, additional tests have been performed with the newly on-boarded use-cases, (including cross-models between different use-cases). Moreover, scalability tests have been performed regarding the monitoring functionality/capabilities.
Test 4 Component Testing	**Description:** Verification of Real-Time Analytics retrieval of multiple contexts, in terms of: 1. Testing of the ability to process real-time data -whether standalone or aggregated- received from streaming sources and file systems. 2. Testing of deployment of streaming algorithms (e.g. for regression), which can simultaneously learn from the streaming data as well as apply the model on the streaming data. **Success Criteria:** Correct processing of incoming data and implementation of streaming algorithms. **Testbed:** UBITECH/Demonstrators' testbeds.
Result/ Comments	Standalone testing of Real-Time Analytics component has been performed with a test use-case (PPDR) data and artificially generated data as reported in [MATILDA-D6.7].

Test 5 Component Testing	**Description:** Verification of extraction of advanced insights and events from the monitoring process, e.g. through data mining, as well as predictive and prescriptive analytics mechanisms (i.e. regression, clustering or classification algorithms). **Success Criteria:** Successful deployment of specific KPI prediction models based on historical data and measurement of their performance. **Testbed:** UBITECH/Demonstrators' testbeds.
Result/ Comments	Integration of the monitoring process with advanced machine learning processing libraries has been tested for batch analytics, i.e. offline learning using historical data. The capability to measure model performance KPIs, based on the model employed, has been tested as well as reported in [MATILDA –D6.7].

Table 7: Lifecycle Management of a Service Request.

Test Objective	10	**Type**	Functional
Title	Lifecycle management of a service request and high-level orchestration of network and compute resources (OSS operation)		
Validation method – Tests	Tests to be performed include: • the interface between the VAO and the underlying network and compute resources domains; • the high-level orchestration of the creation of the network/compute slices within a domain; • the interaction with the NFVO for the incorporation of NSs in the network slices within a domain; • the management of resources within a domain; • the monitoring of nodes/resources within a domain.		
KPIs	**Success Criteria:** Successful performance of OSS functionalities related to the lifecycle management of a service request and the high-level orchestration of network and compute resources. Consistency maintained between the OSS information and the actual provisioning of the requested service.		
Components	OSS, NFVO, VIMs.		
Testbed	All.		

Detailed Tests Description: OSS Operation Testing	
Test 1 Component Testing & Functionality Evaluation	**Description**: Verification of the interface between the VAO and the underlying network and compute resources domains in terms of consistency and correctness of mapping/cross-checking of service graphs' resource requirements to resource requests accounts across a single domain or multiple domains. Verification of consistency maintained between the VAO originated application requests and the user account privileges/services (maintained in user SLAs).

	Success Criteria: Correct mapping between the slice intent provided by the VAO and the resources requested to the VIM(s) and NFVO. Consistency maintenance/resolution between the VAO originated application requests and the user account privileges/services (maintained in user SLAs). **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds
Result/ Comments	Verification of consistency between slice intent requirements and selected computing resources demonstrated during the demo in November 2019 in Ljubljana. Correspondence between resources (in terms of RAM/disk space, etc.) allocated for the VMs and what required in the slice intent. NS provisioning demonstrated during the Ljubljana and Bucharest demos.
Test 2 Component Testing	**Description**: Verification of the high-level orchestration of the creation of the network/compute slices within a domain. Verification of keeping track of requests for slices from VAO, of provisioned slices and resources' offers, along with user account privileges/services information. **Success Criteria:** A repository keeps track of the required user and status information. Testbed: CNIT/UBITECH
Result/ Comments	Mongo DB has been adopted to as the Persistency Layer component for maintaining configuration parameters. The figures below show the slice intent requests received from the VAO (Figure 3). As an example, we can consider the slice with ID 53, composed of four nodes each characterized by a number of constraints. For ease of readability, the screenshot shows the constraints of one node (Figure 4). Additional information stored in the DB regard the links composing the application graph (Figure 5) and the authentication details (Figure 6). This information is aligned with the one maintained in the VAO (Figure 7) .Finally, the repository keeps track of the materialized slices as well (Figure 8), including the VIMs hosting the application graph's components.

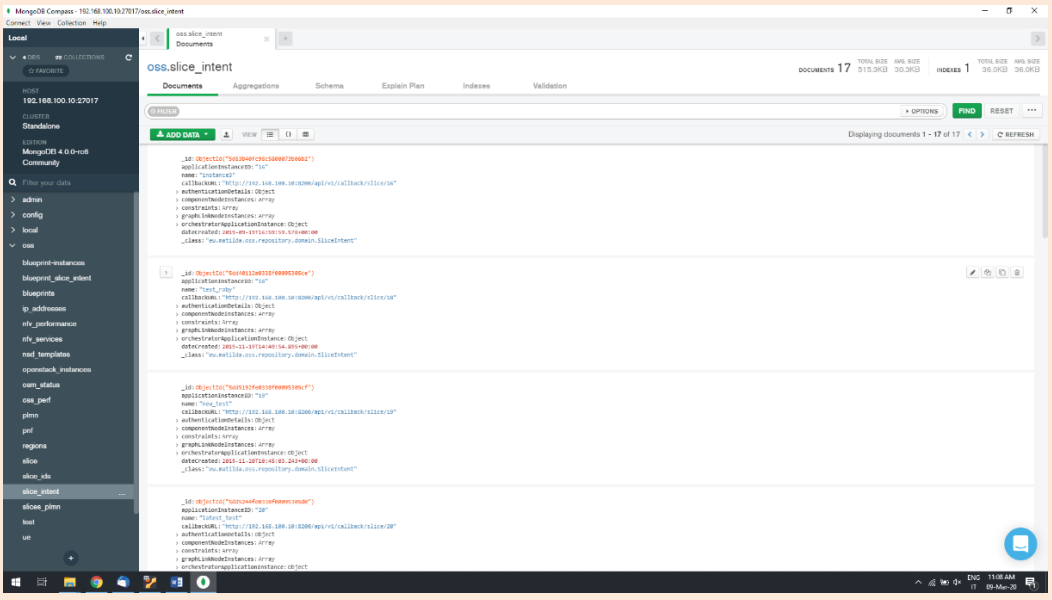


Figure 3: List of slice intents stored in the Persistency Later (MongoDB).

```
componentNodeInstances: Array
  0: Object
     componentNodeInstanceID: "67"
     componentNodeInstanceName: "PPDRPhpDashboard92LB"
  1: Object
  2: Object
  3: Object
constraints: Array
  0: Object
     constraintID: "338"
     componentNodeInstanceID: "67"
     constraintMetric: "MIN_V_CPU"
     constraintUnit: "amount"
     constraintValue: "2"
     category: "COMPONENT_HOSTING"
     type: "HARD"
  1: Object
     constraintID: "337"
     componentNodeInstanceID: "67"
     constraintMetric: "MIN_RAM"
     constraintUnit: "mb"
     constraintValue: "2048"
     category: "COMPONENT_HOSTING"
     type: "HARD"
  2: Object
     constraintID: "335"
     componentNodeInstanceID: "67"
     constraintMetric: "MIN_WORKERS"
     constraintUnit: "amount"
     constraintValue: "1"
     category: "COMPONENT_HOSTING"
     type: "HARD"
  3: Object
     constraintID: "336"
     componentNodeInstanceID: "67"
     constraintMetric: "MIN_STORAGE"
     constraintUnit: "gb"
     constraintValue: "20"
     category: "COMPONENT_HOSTING"
     type: "HARD"
  4: Object
     constraintID: "334"
     componentNodeInstanceID: "67"
     constraintMetric: "MAX_WORKERS"
     constraintUnit: "amount"
     constraintValue: "5"
     category: "COMPONENT_HOSTING"
     type: "HARD"
```


Figure 4: Example of application graph constraints as reported in MongoDB.


```
graphLinkNodeInstances: Array
  0: Object
     graphLinkNodeInstanceID: "61"
     fromComponentNodeInstanceHexID: "aAkY6spxCD"
     fromComponentNodeInstanceID: "67"
     toComponentNodeInstanceID: "65"
     toComponentNodeInstanceHexID: "jQMVDRFN2u"
     type: "CORE"
  1: Object
     graphLinkNodeInstanceID: "60"
     fromComponentNodeInstanceHexID: "jQMVDRFN2u"
     fromComponentNodeInstanceID: "65"
     toComponentNodeInstanceID: "66"
     toComponentNodeInstanceHexID: "Jiz1EWc5gW"
     type: "CORE"
  2: Object
     graphLinkNodeInstanceID: "59"
     fromComponentNodeInstanceHexID: "jQMVDRFN2u"
     fromComponentNodeInstanceID: "65"
     toComponentNodeInstanceID: "64"
     toComponentNodeInstanceHexID: "wmkN5Exqcq"
     type: "CORE"
  3: Object
     graphLinkNodeInstanceID: "ACCESS_97"
     toComponentNodeInstanceID: "67"
     toComponentNodeInstanceHexID: "aAkY6spxCD"
     type: "ACCESS"
```


Figure 5: Example of application graph links as reported in MongoDB.

```
providerAuthenticationDetails: Array
  0: Object
      _id: "1"
      adapterType: "FIFTH_GENERATION_TELCO_PROVIDER"
      username: "telcoprovider"
      password: "71Nv+hltJhj8HYtZQHcelA=="
      proxy: "127.0.0.1:8085"
services: Array
  0: Object
  1: Object
      componentNodeInstanceID: "67"
      componentNodeInstanceHexID: "aAkY6spxCD"
      componentNodeInstanceName: "PPDRPhpDashboard92LB"
      componentNodeID: "233"
      componentNodeHexID: "EtT8LqeByW"
      componentNodeName: "PPDRPhpDashboard92LB"
      providerID: "1"
      image: "traefik:alpine"
      registry: "maestro-public-repository.ubitech.eu"
      sshKey: "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC1uCM+WTrtr82waAez9ZrPnvIyArAVk5..."
    command: Array
    dependsOn: Array
    ports: Array
      loadBalancer: true
      lambdaProxy: false
    monitoringElasticity: Object
    healthCheck: Object
    flavor: Object
    plugins: Array
      dockerUsername: "maestro"
      dockerPassword: "Oh6xTlbsBfg8gdIBuW/21A=="
```


Figure 6: Example of application graph authentication details as reported in MongoDB.

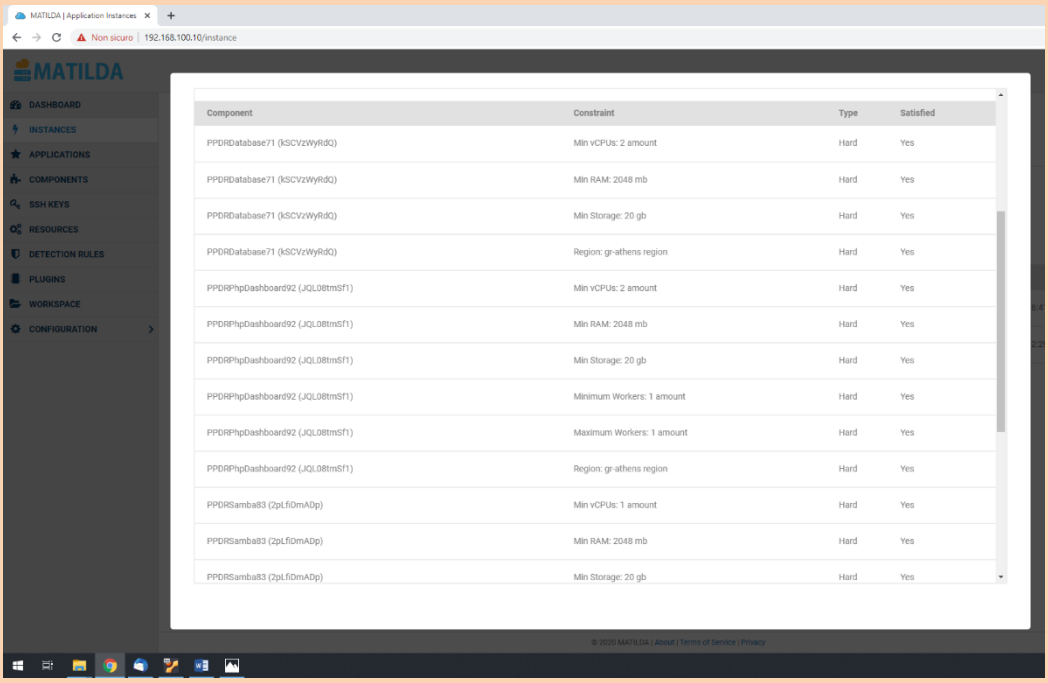


Figure 7: Slice intent details from the VAO GUI.

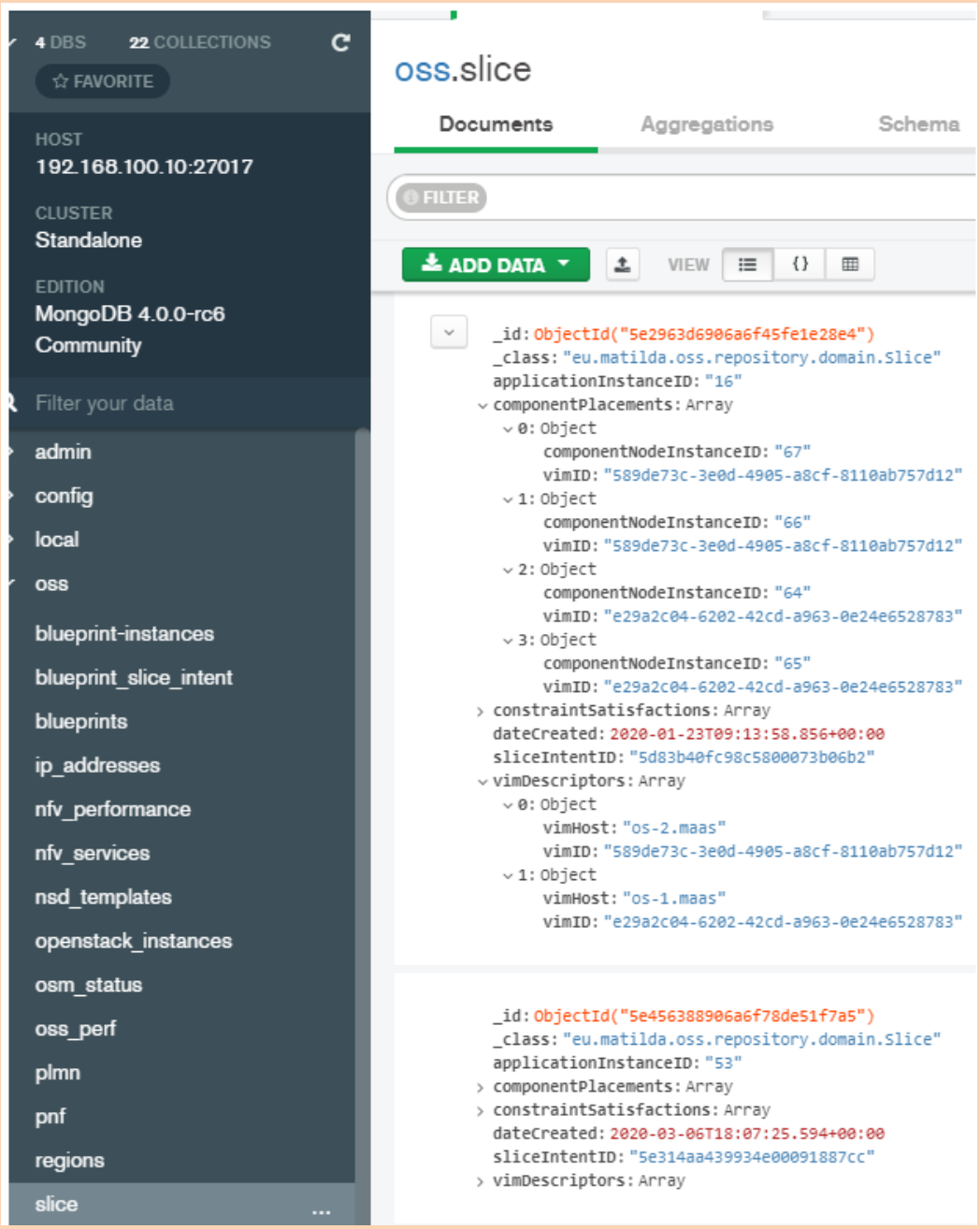
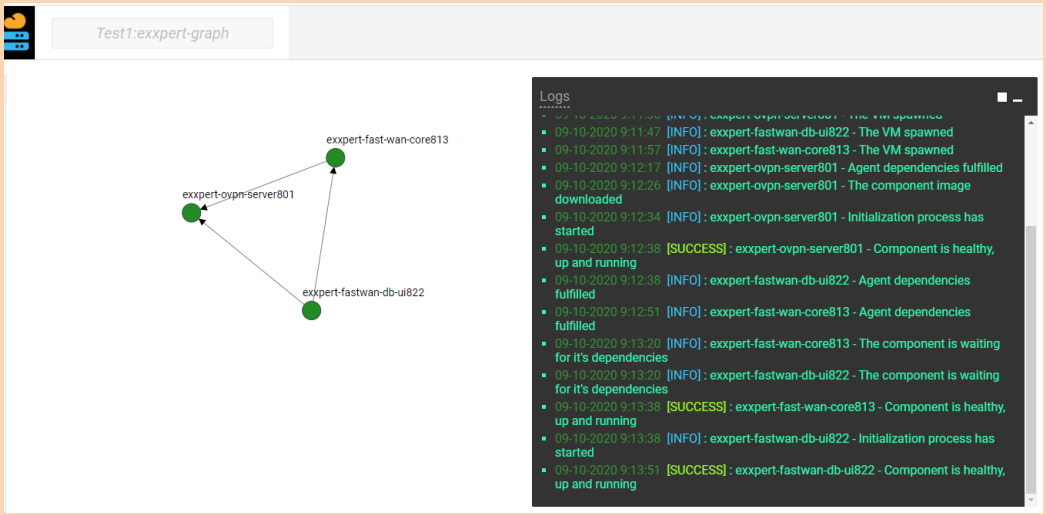
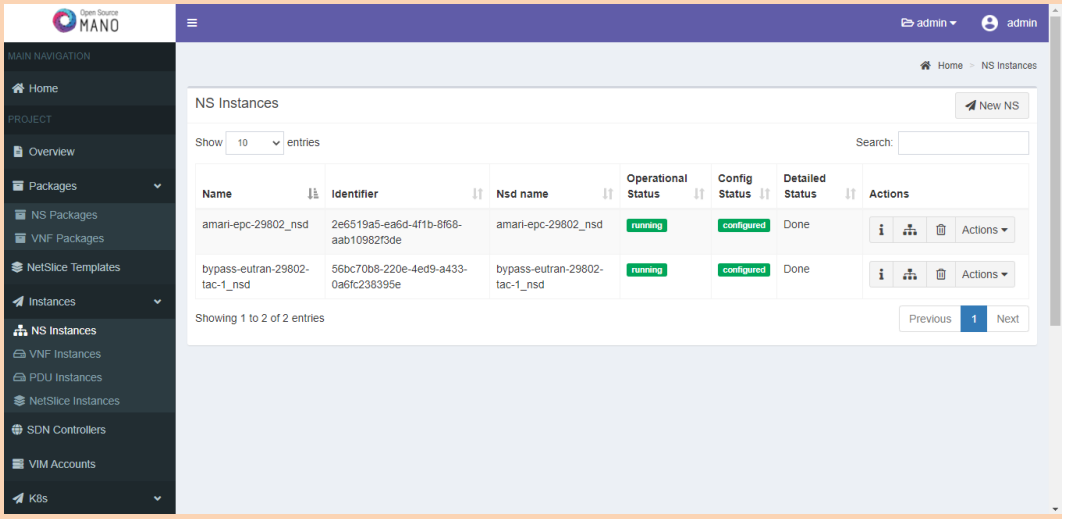
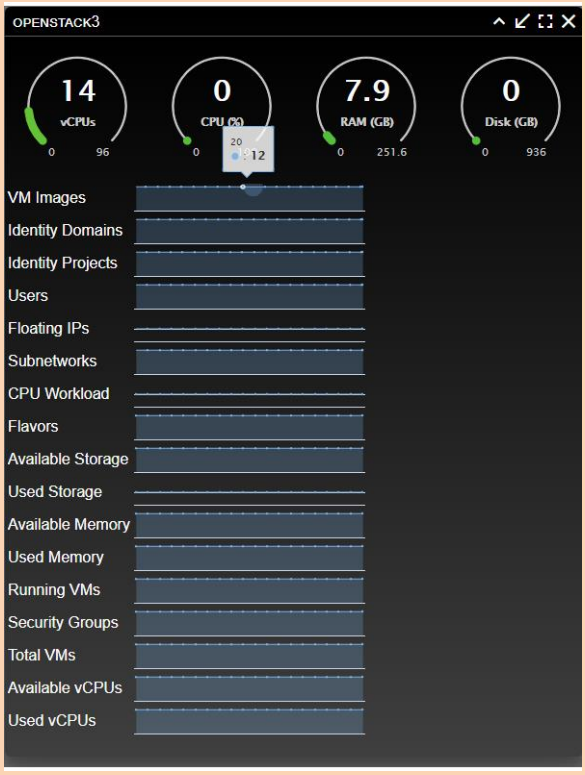


Figure 8: Example of materialized slice as reported in MongoDB. Note the instantiation over multiple domains.

Test 3 Component Testing	**Description**: Verification of the interaction of OSS with the NFVO for the incorporation of VNFs and NSs in the network slices extracted from the slice intent received from the VAO (within a domain). OSS shall request the instantiation of NSs upon resolution of slice intent. **Success Criteria:** Correct resolution of slice intent in terms of identifying required VNFs. Correct communication with NFVO, and correct instantiation of NSs. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds
Result/ Comments	The example materialized slice shown in Figure 9 requires the sole presence of the Public Land Mobile Network (PLMN) network service in Figure 10, which shows that the VNFs composing the service are up and running in OSM.  **Figure 9: Materialized slice intent as shown in the VAO.**  **Figure 10: Deployed network service shown in in the OSM GUI.**

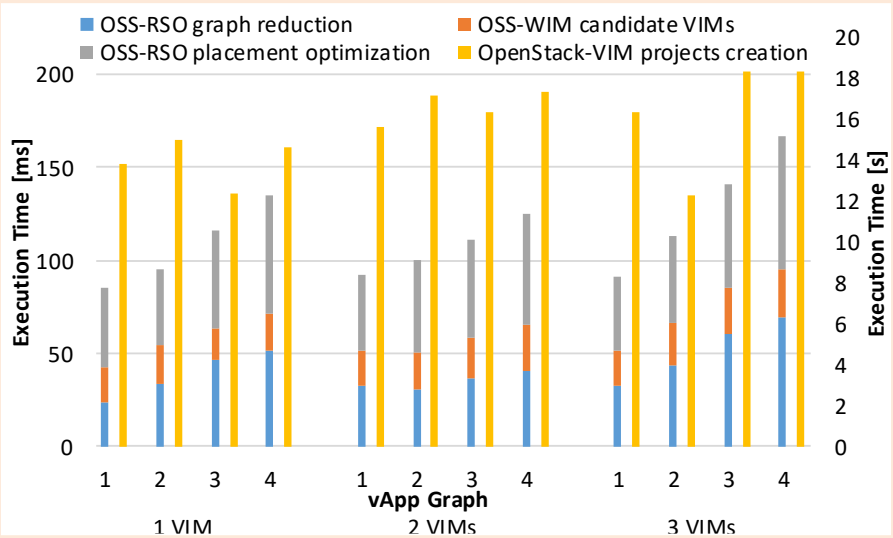
Test 4 Component Testing	**Description**: Verification of the management of resources within a domain including also the monitoring of nodes/resources within this domain. Verification of monitoring of resources in terms of availability of compute nodes and corresponding network links/connectivity. **Success Criteria:** Correct monitoring of compute nodes resources. Correct monitoring of network links/connectivity resources. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds
Result/ Comments	The three submodules composing the RSO, namely Graph Reduction, Utilization Forecasting and Placement Optimization, have been completed and integrated to i) aggregate application components into "macro nodes" according to a set of link QoS constraint thresholds, generating a reduced graph composed of "macro nodes" treated as inseparable in the following placement, ii) periodically collect performance metrics for each VIM registered in the OSS, and iii) select the most suitable VIMs where to place the macro nodes according to both the slice resource requirements and the forecasted metrics. The Utilization Forecasting defines the set of metrics to be used in the selection of the most suitable VIMs by considering the monitoring metrics on vCPU, RAM and disk utilization of the candidate VIMs available in Prometheus. The time series of Prometheus are made available in the OSS GUI, shown in Figure 11, which plots the current parameters as well as a time window frame with a dedicated panel for each compute node.  **Figure 11: Monitoring of compute node resources made available in the OSS GUI.**

Test 5 Component Testing	**Description**: Verification of the high-level orchestration of the creation of the network/compute slices over multiple domains. Verification of keeping track of requests for slices from VAO, of provisioned slices, and resources' offers along with user account privileges/services information. **Success Criteria:** A repository keeps track of the required user and status information for multiple domains. **Testbed:** CNIT/UBITECH CNIT/UBITECH/Demonstrators' Testbeds.
Result/ Comments	Please refer to Test 2.
Test 6 Component Testing	**Description**: Verification of the management of resources within a domain including also the monitoring of nodes/resources within multiple domains. Verification of monitoring of resources in terms of availability of compute nodes and corresponding network links/connectivity. **Success Criteria:** Correct monitoring of compute nodes resources. Correct monitoring of network links/connectivity resources. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds.
Result/ Comments	Please refer to Test 4.

Table 8: Lifecycle Management of Slices.

Test Objective	11	**Type**	Functional
Title	Lifecycle Management of Slices		
Validation method – Tests	Tests to be performed are related to the lifecycle management of slices on the infrastructure, and include: • the initial provisioning of the slice (in terms of compute/network resources and QoS) requested from the slice intent for a specific application deployment; • the deletion of the slice (release of resources) upon application instance termination.		
KPIs	**Success Criteria:** Successful performance of functionalities related to the lifecycle management of a slice.[2] Consistency maintained between the Slice Manager information and the actual slice state.		
Components	Slice Manager (as part of the OSS, and Resource Selector Optimizer).		
Testbed	All.		

[2] Tests related to the successful performance of functionalities referring to the lifecycle management of a slice are covered in the tests defined in Table 5, Table 9, Table 10, Table 11, and Table 12. So, tests related to this objective will focus on testing the consistency maintained between the Slice Manager information and the actual slice's state.

Detailed Tests Description: Lifecycle Management of Slices Testing	
Test 1 Component Testing & Functionality Evaluation	**Description:** Verification that upon instantiation of a slice the relevant information is maintained at the Slice Manager side. **Success Criteria:** Consistency maintained between the Slice Manager information and the actual slice state. **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Instantiate Network Slice** • Time to Instantiate Network Slice. **Testbed:** CNIT/UBITECH.
Result/ Comments	Results in Figure 12 show the distribution of the execution times ascribable to the OSS (left bar, stack of the RSO and WIM execution times) and to OpenStack to instantiate a network slice, which completes the Day-1 operations of the vApp lifecycle. Execution times are plotted over the application graph and the number of VIMs available during the test, and are reported in milliseconds for the OSS and in seconds for OpenStack. The OSS execution times grow with the size of the application graph because of the contribution of both RSO phases. In fact, it can be seen that the execution time ascribable to the WIM is constant with the application graph size and only presents negligible variations with the number of VIMs. On the other hand, since OpenStack performs the same operations regardless of the application size, and each VIM creates its project independently, the execution times are not correlated neither with the application graph size nor with the number of available VIMs. These results highlight how the execution times ascribable to OpenStack represent over 99% of the total, bringing the total deployment time in a range that is suitable for the IaaS context, but is definitely not acceptable for the 5G vertical environment, in which service agility is seen as a non-negotiable.  **Figure 12: Total execution times for Day-1 operations ascribable to the OSS and to OpenStack.**

Test 2 Component Testing & Functionality Evaluation	**Description:** Verification that upon deletion of a slice the relevant information is maintained at the Slice Manager side. **Success Criteria:** Consistency maintained between the Slice Manager information and the actual slice state. **Evaluation KPI:** **5G-PPP Phase 4. Terminate >> Terminate Network Slice** • Time to Terminate Network Slice **Testbed:** CNIT/UBITECH
Result/ Comments	Every time a slice is materialized, information related to the time required for each phase of the materialization is maintained in Mongo DB. The collection shown in Figure 13 reports this information, which was use, for example, to characterize the execution times for Day-1 operations in Figure 12. Regarding the time to terminate a network slice, the collection below shows that the time required to release all slice resources is around 14 s, and the complete deletion of the slice takes only a few more microseconds.

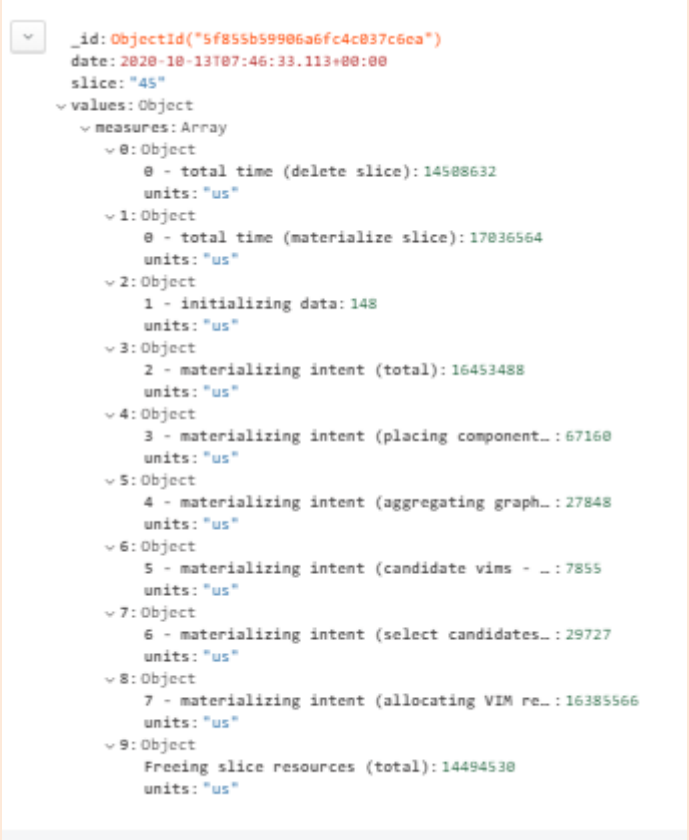


Figure 13: Collection maintained in the Mongo DB regarding slice materialization.

Table 9: Management of VNFs/NSs.

Test Objective	12	**Type**	Functional
Title	Management of NSs (mainly with regard to VNFs)		
Validation method – Tests	Tests to be performed are related to the lifecycle management/support of network functions, including: • the mapping of the slice intent to specific network resources and VNFs; • the re-use and configuration of VNFs from multiple tenants / in multiple slices; • the instantiation of VNFs for a specific slice; • the termination of the slice and associated VNF instances' operation.		
KPIs	**Success Criteria:** Successful performance of the functionalities that are related to the VNFs lifecycle. Consistency maintained between the NFVO information and the actual instantiated VNFs' state.		
Components	NFVO.		
Testbed	All.		

Detailed Tests Description: Management of NSs	
Test 1 **Component Testing & Functionality Evaluation**	**Description:** Verification of the mapping of the slice intent to specific, suitable 3GPP NSs, to support the 5G-ready applications' deployment. 3GPP network services are mainly related to RAN, composed of the PNFs constituting the eNBs, the EPC functional entities, and the additional VNF providing MEC functionality, for defining bearers on a per-UE basis. **Success Criteria:** Correct mapping of the slice intent to specific, suitable 3GPP network services (bearer). **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Instantiate & Activate Network Service** • Time to Instantiate & Activate Network Service. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds.
Result/ Comments	In order to realize slicing over 4G networks, a number of ad-hoc solutions have been implemented, of which an accurate description can be found in D4.2 [MATILDA-D4.2]. As shown in Figure 14(and previously on the OSM GUI in Figure 10), the PLMN is composed of two VNFs, one realizing the S1 bypass and the other the monolithic EPC.

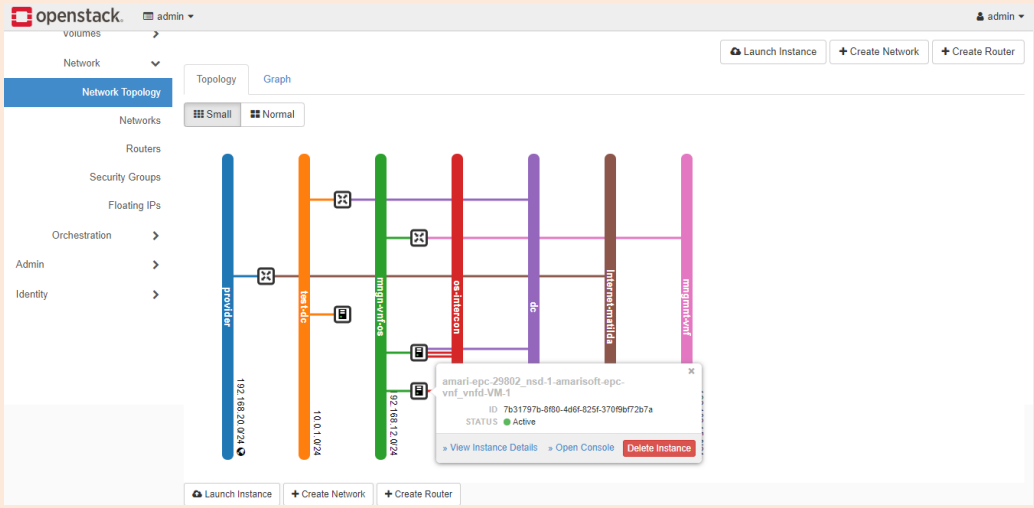


Figure 14: Topology of the PLMN service as shown in OpenStack.

Regarding the time to instantiate and activate a network service, measurements have been performed on the time required to complete each Day-0/1/2 operation composing the lifecycle of the PLMN NS.

Results represent the time ascribable to each individual operation that takes place at the initial bootstrap of the MATILDA platform. In more details, the NFVSM identifies the required NS and selects the related blueprint (operation indicated as 0-1 in Figure 15). Starting from the information in the blueprint, the NFVSS is responsible to properly generate NSD packages (operation 0-2) and onboard them on the NFVO, which concludes Day-0 operations (0-3).

In Day-1 operations, the NFVSS asks the NFVO to instantiate the NSIs (1-1) and waits for the completed instantiations. Day-1 operations end when all VNF images and their corresponding VNF Manager (VNFM, in the form of a Juju charm) are up and running (1-2).

Upon successful fulfilment of Day-0/1 operations, the NFVSS retrieves the proper VNF Configurators (2-1) and sends the list of Day-2 primitives to the Juju charm of each VNF composing the service. In the results, start (2-2, 2-3) and stop (2-4, 2-5) primitives have been configured for the two VNFs. Day2 operations end once these primitives have been successfully run for each service (2-6).

The operation involving OSM, e.g., the instantiation of the NSIs (1-1), accounts for almost all the execution time, keeping the whole lifecycle in the order of magnitude of minutes.

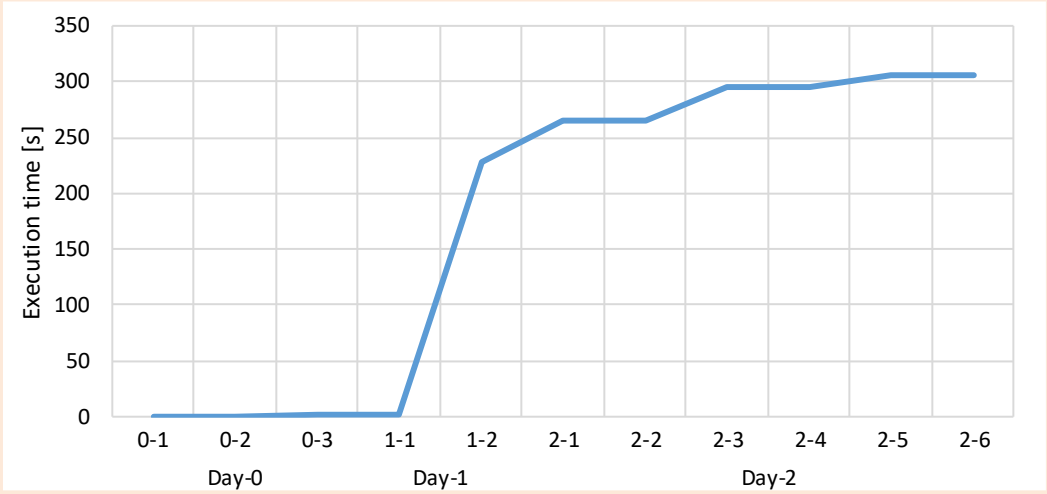
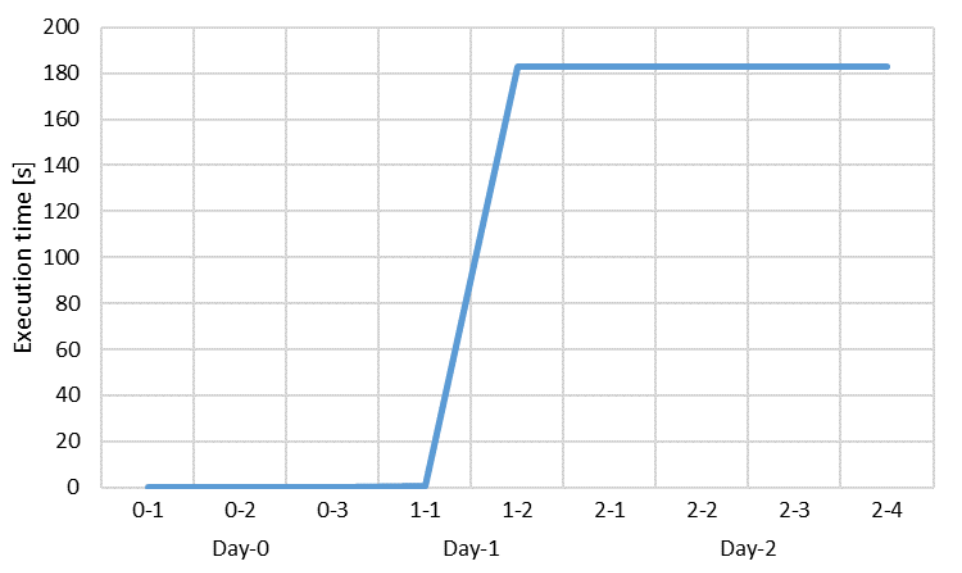


Figure 15: Total execution times for Day-0/1/2 operations on a 3GPP NSI ascribable to the OSS and to OSM.

Test 2 Component Testing & Functionality Evaluation	**Description:** Verification of the mapping of the slice intent to specific, suitable NSs, including non-3GPP network VNFs. Non-3GPP VNFs can include security-related, traffic handling, and monitoring VNFs. **Success Criteria:** Correct mapping of the slice intent to specific, suitable VNFs. **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Instantiate & Activate Network Service** • Time to Instantiate & Activate Network Service. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds.
Result/ Comments	The results plotted in Figure 16 represent the execution times measured for Day-0/1/2 operations on a Non-3GPP NSI. The considered NS is a virtual router based on the VyOS project that provide auxiliary network functionalities on a per vApp slice basis, such as routing protocols, firewall, DNS, DHCP, NAT, etc. Results represent the time ascribable to each individual operation that takes place upon reception of a slice intent to put into operation the network slice. In more details, the NFVSM identifies the required NS and selects the most suitable blueprint fulfilling the QoS requirements (operation indicated as 0-1 in Figure 16). Starting from the information in the blueprint, the NFVSS is responsible to properly generate NSD packages (operation 0-2) and onboard them on the NFVO, which concludes Day-0 operations (0-3). In Day-1 operations, the NFVSS asks the NFVO to instantiate the NSIs (1-1) and waits for the completed instantiations. Day-1 operations end when all VNF images and their corresponding VNF Manager (VNFM, in the form of a Juju charm) are up and running (1-2). Upon successful fulfilment of Day-0/1 operations, the NFVSS retrieves the proper VNF Configurators (2-1) and sends the list of Day-2 primitives to the Juju charm of each VNF composing the service. In the results, start (2-2) and stop (2-3) primitives have

	been configured for this NS. Day2 operations end once these primitives have been successfully run for each service (2-4). The operation involving OSM, e.g., the instantiation of the NSIs (1-1), accounts for almost all the execution time, keeping the whole lifecycle in the order of magnitude of minutes.  **Figure 16: Total execution times for Day-0/1/2 operations on a non-3GPP NSI ascribable to the OSS and to OSM.**
Test 3 Component Testing & Functionality Evaluation	**Description:** Verification of the re-use, configuration and instantiation of VNFs in multiple slices. **Success Criteria:** Correct initial configuration and instantiation of VNFs from multiple tenants / in multiple slices. **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Instantiate & Configure VNFs in Service Chain** • Time to Instantiate & Configure VNFs in Service Chain. **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds.
Result/ Comments	The time required to instantiate and configure a VNF can be deduced from Test 2, as they consider a service composed of a single VNF, by accounting only for Day-0/1 operations. Actually, the difference is quite negligible, as the completion of Day-1 operations takes 182.819 s against the 182.989 s required for Day-2. This result, along with Tests 1 and 2, show how the algorithms operating in the OSS have negligible execution times, while OSM is responsible for around 99% of the deployment of NSs, bumping their lifecycle up to the order of magnitude of minutes. This range is adequate for the IaaS environment, but OSM is still not suitable to fulfil 5G requirements.

Test 4 Component Testing & Functionality Evaluation	**Description:** Verification of termination of VNFs once the relevant network slice is deleted. **Success Criteria:** Termination of VNFs and release of resources once the relevant network slice is deleted. **Evaluation KPI:** **5G-PPP Phase 4. Terminate >> Terminate VNFs in Service Chain** • Time to Terminate VNFs in Service Chain **5G-PPP Phase 4. Terminate >> Terminate Network Service** • Time to Terminate Network Service **Testbed:** CNIT/UBITECH/Demonstrators' Testbeds
Result/ Comments	The termination of a network service requires times in the order of magnitude of one minute, while the current implementation of OSM as integrated in MATILDA does not allow to terminate individual VNFs.

Table 10: Management of Infrastructure Resources.

Test Objective	13	**Type**	Functional
Title	Management of Infrastructure Resources		
Validation method – Tests	Tests to be performed are related to management of infrastructure resources, including: • exposing of infrastructure resources' availability to the necessary entities including the CSM/VAO and NFVO, • instantiation of infrastructure resources upon request after resolution of resources' availability, • multi-tenancy.		
KPIs	**Success Criteria:** Successful performance of the functionalities that are related to the infrastructure resources' management on PoPs.		
Components	VAO, VIM.		
Testbed	All.		

Detailed Tests Description: Management of Infrastructure Resources	
Test 1 Component Testing	**Description:** Verification of exposure of infrastructure resources' availability to the necessary entities including the CSM/VAO and NFVO. **Success Criteria:** Infrastructure resources' availability information exchanged between VIM and NFVO and CSM/VAO is correct. **Testbed:** CNIT/UBITECH.
Result/ Comments	Information exchange between OSS and VIM has been successfully tested and demonstrated during the demo in November. Regarding the NFVO, the communication towards the OSS are handled by the NFV Convergence Layer (NFVCL) [MATILDA-D4.2].

As an example of the information exchange between the OSS and the NFVCL please refer to Figure 17. While the PLMN is being instantiated following the steps presented in Figure 15, information about the selected blueprint and the corresponding NSD are printed on screen. This information corresponds to the one available on the OSM GUI in Figure 18. Once the instantiation process is completed, the OSS keeps track of the service in Mongo DB, see Figure 19.

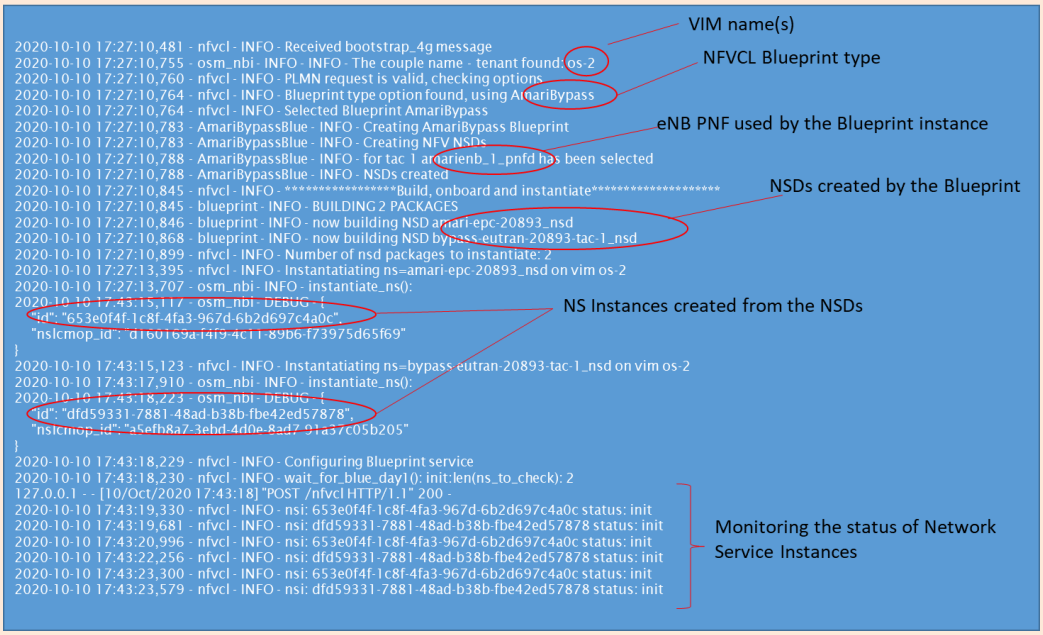


Figure 17: Output produced by the NFVCL during the instantiation of a NS.

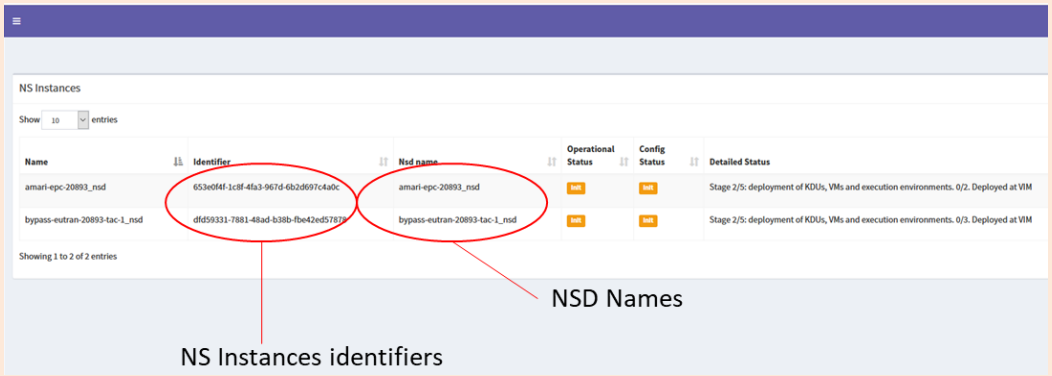


Figure 18: OSM CLI showing the NSIs being created.

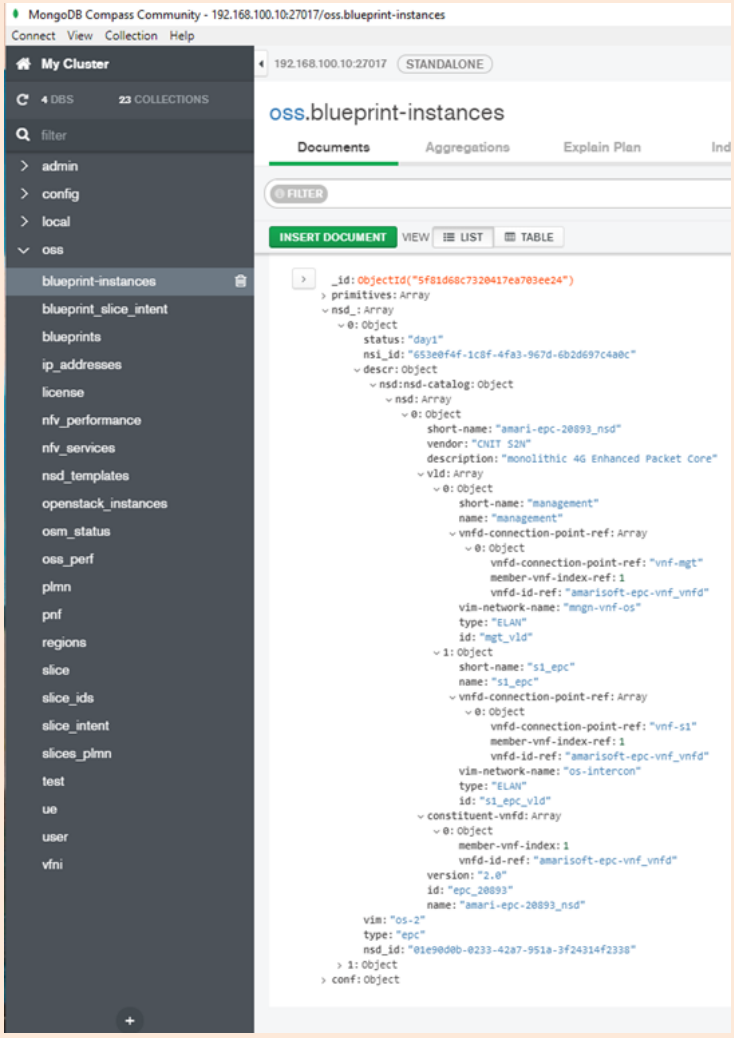


Figure 19: Information on the instantiated NS maintained in Mongo DB.

Test 2 Component Testing & Functionality Evaluation	**Description:** Verification of instantiation of appropriate infrastructure resources upon request after resolution of slice intent and resources availability. **Success Criteria:** Correct instantiation of appropriate infrastructure resources upon request. **Testbed:** CNIT/UBITECH.
Result/ Comments	Please refer to Table 6, Test 1.
Test 3 Component Testing	**Description:** Verification of multi-tenancy support, isolation and sharing of infrastructure resources. **Success Criteria: (5GPPP KPI/Success Criterion: Multi-tenancy)** Multiple tenants can share infrastructure resources. **Testbed:** CNIT/UBITECH.
Result/ Comments	Please refer to Table 8, Test 3.

Table 11: Management of Wide-area Network Resources.

Test Objective	14	**Type**	Functional
Title	Management of Wide-area Network Resources		
Validation method – Tests	Tests to be performed are related to the logical interconnectivity among sets of service/application components instantiated in different PoPs, including: • the network resources allocation and QoS provisioning for each link defined in the application service graph; • the maintenance and modification of the network resources based on runtime policies and general status of the WAN; • the release of network resources upon termination of the application instance operation.		
KPIs	**Success Criteria:** Successful performance of the functionalities that are related to the network resources provisioning lifecycle.		
Components	WIM.		
Testbed	All.		

Detailed Tests Description: Management of Wide-area Network Resources	
Test 1 Component Testing	**Description:** Verification that the VIM has calculated a list of candidate sets of PoPs for deployment of the groups of service/application obtained from the reduced service graph. **Success Criteria:** The ENM GUI shows the list of candidate VIM(s) fulfilling the proximity requirements. **Testbed:** CNIT.
Result/ Comments	The Ericsson WIM solution has been deeply tested with multiple testing conditions and the capability to calculate the proper sets of PoPs and to effectively show the list of candidate VIMs fulfilling the proximity requirements to the ENM GUI has been successfully validated.

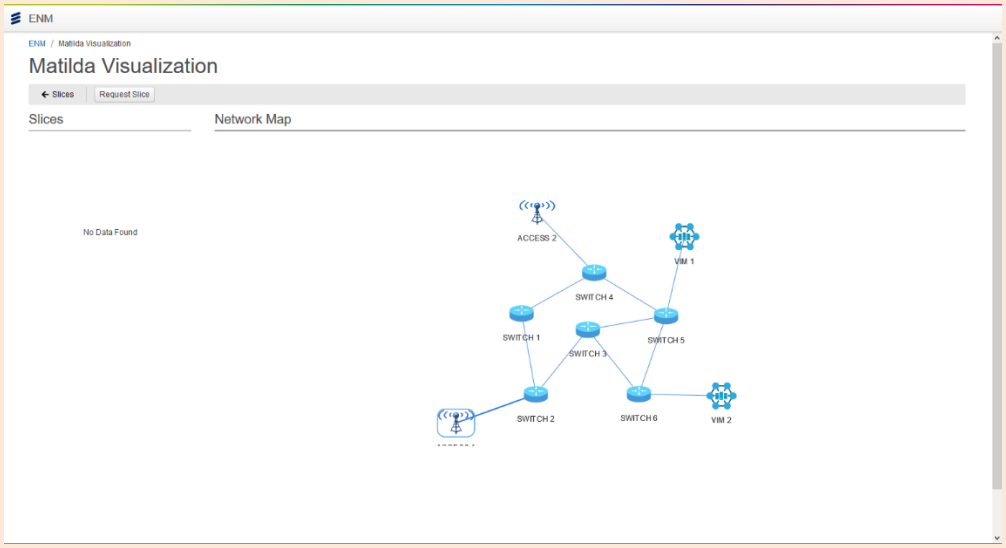


Figure 20: Matilda Visualization – Network Topology.

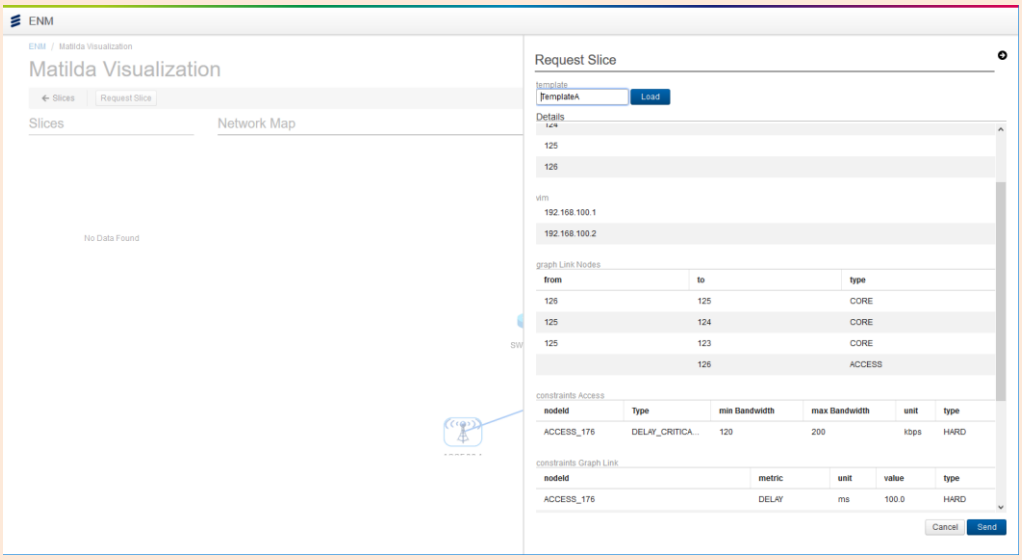


Figure 21: Matilda Visualization – Slice Request.

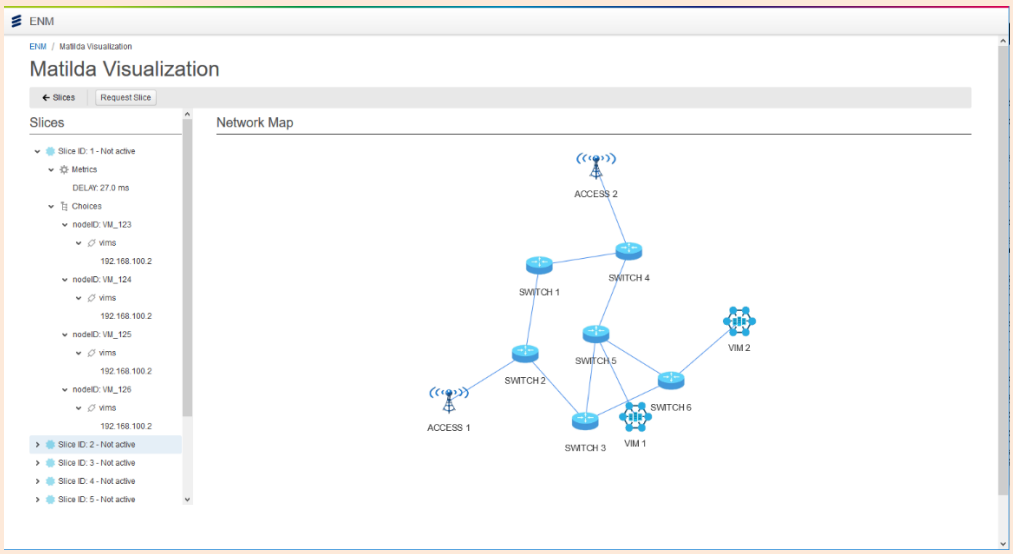


Figure 22: Matilda Visualization – Compute Slice Response.

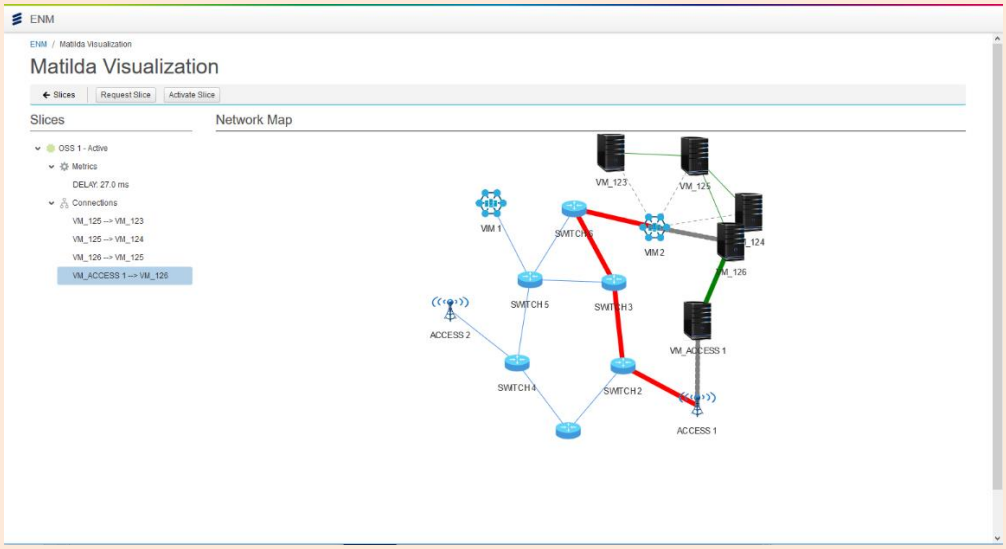


Figure 23: Matilda Visualization – Slice Activation and Routing Setup.

Test 2 Component Testing & Functionality Evaluation	**Description:** Test and evaluation of the computational times and scalability effectiveness in relation to the elaboration of the list of candidate sets of PoPs for deployment of the groups of service/application obtained from the reduced service graph. **Evaluation KPI:** • Fast computational time to elaborate the list of candidate sets of PoPs. **Testbed:** CNIT.

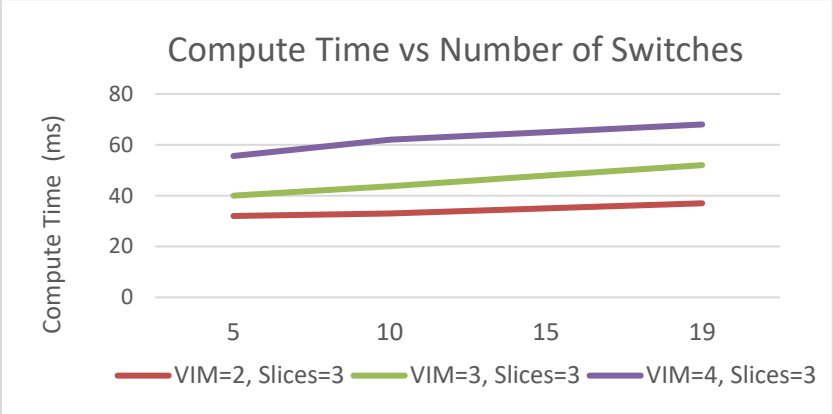


Figure 24: Compute Time vs Number of Switches.

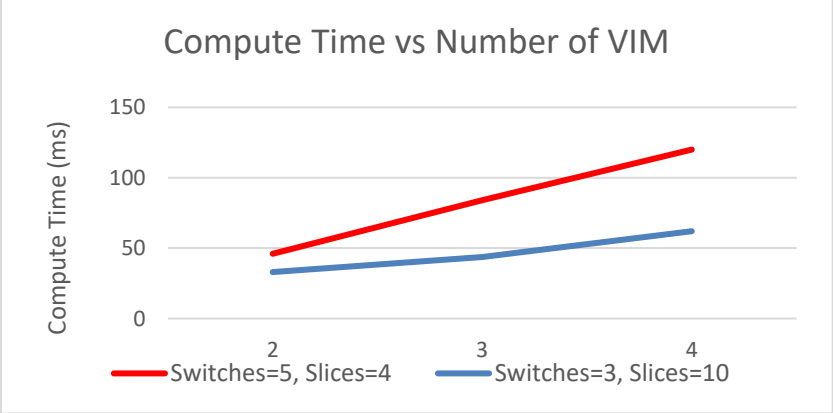


Figure 25: Compute Time vs Number of VIM.

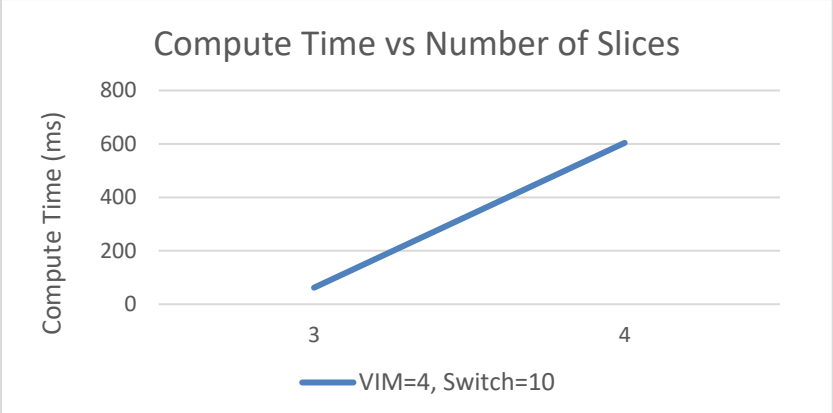
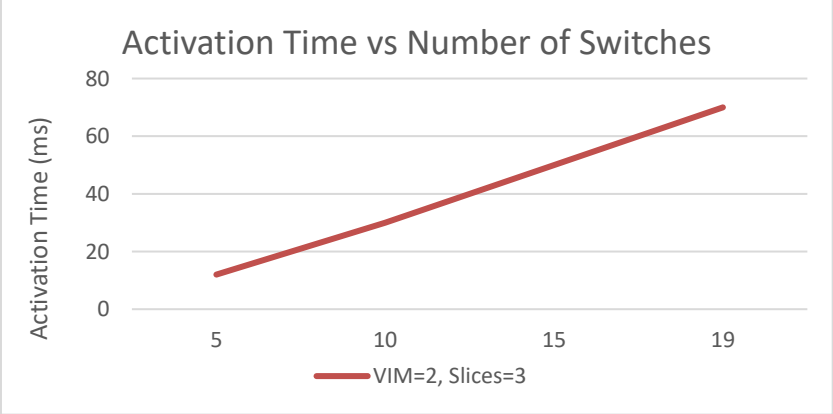
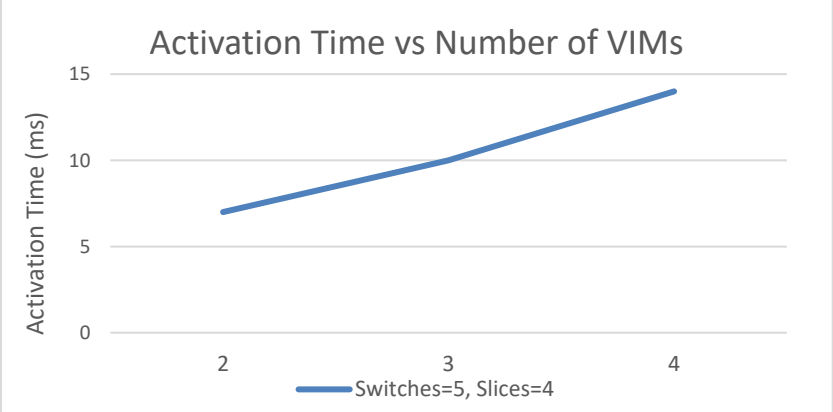


Figure 26: Compute Time vs Number of Slices.

Test 3 Component Testing & Functionality Evaluation	**Description:** Verification of establishment of interconnectivity among sets of service/application components instantiated in different PoPs, by provisioning the correct network resources and QoS for each link defined in the application service graph. **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Configure SDN Infrastructure** • Time to Configure SDN Infrastructure (OpenFlow). • Fast activation time. **Success Criteria:** Connectivity achieved as slice instantiation based on the application slice intent, and network resources management/availability resolution. **Testbed:** CNIT
Result/ Comments	 **Figure 27: Activation Time vs Number of Switch (VIM=2, Slice=3).**  **Figure 28: Activation Time vs Number of VIMs (Switches=5, Slices=4).**

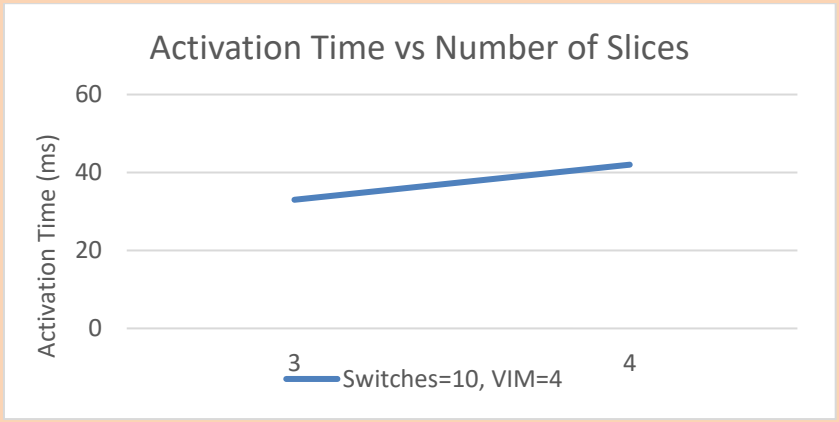


Figure 29: Activation Time vs Number of Slices (Switches=10, VIM=4).

Table 12: Multi-site Resource Management.

Test Objective	15	**Type**	Functional
Title	Multi-site Resource Management		

Validation method – Tests	Tests to be performed are related to managing resources at diverse facilities, like central/remote public/private/hybrid cloud facilities or at the mobile network edge, more specifically including:
	• information exchange between the VAO (Execution Manager), the Computing Slice Broker and the VIMs regarding their availability of resources;
	• deployment of applications/application components at Network Service Provider's edge facilities using information related to:
	  • end-user location and locality of computing resources,
	  • availability of resources of various PoPs.
	• lifecycle management of an application component deployed at the Telecom Service Provider's facilities through (evolved) IaaS/PaaS APIs.		

KPIs	**Success Criteria:** Successful performance of the functionalities that are related to managing resources at diverse PoPs. Consistency maintained between the information at the Multi-Site Resource Manager, and the actual PoPs and Network Infrastructures.		
Components	VAO, Multi-site Resource Management functionality, VIM, WIM.		
Testbed	CNIT		

Detailed Tests Description: Multi-site Resource Management	
Test 1 Component Testing	**Description:** Verification that the correct information is exchanged between the VAO's Execution Manager, the Computing Slice Broker and a number of VIMs (of various PoPs, like central/remote public/private/hybrid cloud facilities or at the mobile network edge), regarding their availability of resources.

	Success Criteria: Exchange of the necessary and correct information regarding the resources' request and availability. **Testbed:** CNIT, UBITECH and/or Demonstrators' facilities.
Result/ Comments	The correct information exchange has demonstrated during the Ljubljana and Bucharest demos.
Test 2 Component Testing & Functionality Evaluation	**Description:** Verification of deployment of applications/application components at Network Service Provider's edge facilities using information related to locality of computing resources and availability of resources of various PoPs. **Success Criteria:** Correct placement of compute resources to the appropriate PoP after resolution of slice intent information. **Evaluation KPI:** **5G-PPP Phase 2. Instantiate, Configure & Activate >> Instantiate & Configure MEC Application** • Time to Instantiate & Configure MEC Application. **Testbed:** CNIT, UBITECH and/or Demonstrators' facilities.
Result/ Comments	For the placement of compute resources, please refer to Table 6, Test 1. For the KPI, please refer to Figure 12.

5 User Process and Performance Evaluation of MATILDA Solution

Complete demonstrations have been made with all MATILDA vertical applications, and the experimentation, testing and evaluation results on a per demonstrator/ vertical application case have been detailed in MATILDA Deliverables [MATILDA-D6.8]-[MATILDA-D6.12]. This section summarises these results, and serves as concluding information from all this work. Readers of this section that want to find more details about the applications, the testing procedures, and the results shall refer to these documents.

5.1 On-boarding Process Evaluation Aspects

With the completion of the transformation of the stand-alone vertical applications (that are available in the context of the MATILDA project) into 5G-ready applications and the on-boarding process of these applications' graphs, significant hands-on experience has been acquired by the verticals as MATILDA end-users. Given this experience, the MATILDA performance evaluation tests related to the on-boarding process have been refined and specified at the level of specific tests' and success criteria, in the tables of this section. Final results obtained from the verticals'/end-users' perspective are also summarised in these tables.

Table 13: User Friendliness Evaluation.

Test Objective	16	**Type**	Other
Title	User Friendliness		
Relevant UCs	All.		
Validation method – Tests	User friendliness of the MATILDA solution interfaces to users/stakeholders can be evaluated at MATILDA system design and development phases by the end-users represented by the relevant consortium partners.		
KPIs	**Relevant KPIs**: Evaluation feedback collected by various end users/stakeholders of the MATILDA solution.		
Components	General Solution.		
Testbed	All.		

Detailed Tests Description: User Friendliness	
Test 1	**Description:** Evaluation of User Friendliness of the on-boarding process of the vertical applications. **Success Criteria:** >90-100% of end-users find the on-boarding process user-friendly (feedback received over the project period through discussions between partners as well as on the basis of a questionnaire with regard to quality of graphics, responsiveness of interface, identification of steps and navigation through screens, etc. The end-user questionnaire was distributed and discussed about towards the project end).

Result/ Comments	A recursive Software Engineering process has been adopted throughout the project so that any comments/remarks from verticals has been taken into account and addressed so that the user interface meets their requirements. Results have shown that the on-boarding process of vertical applications is considered user-friendly for the partners/personnel involved in the project.
Test 2	**Description:** Evaluation of understandability of requested input for the description of resources/performance requirements, etc., during the on-boarding process of the vertical applications. **Success Criteria:** >90-100% of end-users can clearly understand the input that is needed during the on-boarding process (feedback received over the project period through discussions between partners, as well as on the basis of a questionnaire).
Result/ Comments	A recursive Software Engineering process has been adopted so that any questions/clarifications asked from verticals have been taken into account and addressed, so that the user interface becomes more understandable, and supportive documentation has been generated for the same purpose. Results have shown that the parameters requested during the on-boarding process of vertical applications is easy to understand, with the help of supportive documentation.
Test 3	**Description:** Evaluation of completeness/correctness of description of required performance/ resources. **Success Criteria:** • 100% of end-users find the description of required performance/ resources in terms of parameters, target values, etc., requested during the on-boarding process correct in terms of defining their vertical application. • >90-100% of end-users find the description of required performance/ resources in terms of parameters, target values, etc., requested during the on-boarding process complete in terms of defining their vertical application. (feedback received over the project period through discussions between partners, as well as on the basis of a questionnaire).
Result/ Comments	A recursive Software Engineering process has been adopted so that any addition requested from verticals (also through a questionnaire that has been discussed) has been taken into account and added to the MATILDA VAO. Results have shown that the parameters requested during the on-boarding process describe the partners' vertical applications in a complete and correct way.

5.2 Deployment Process Evaluation Aspects

With the completion of the deployment of the stand-alone vertical applications (that are available in the context of the MATILDA project), significant hands-on experience has been acquired by the verticals as MATILDA end-users. At this stage, key aspect for evaluation has been the time needed for deployment of various modules – as perceived by the end user; thus taking into account the end-user competence in performing these processes. Focusing on this, the MATILDA performance evaluation tests have been refined and specified at the level of

specific tests' and success criteria as follows in this section. Final results obtained from the verticals'/end-users' perspective are also summarised.

Table 14: Speed of Application Deployment Evaluation.

Test Objective	17	**Type**	Other
Title	Speed of Application Deployment		
Relevant UCs	All.		
Validation method – Tests	Tests include measurement and evaluation of the time required for an application component and application deployment for the various UCs', at various test infrastructures, with various deployment parameters to be defined at run-time (service graphs and run-time policies). The factors affecting the speed of deployment in each case will be identified and evaluated.		
KPIs	**Relevant KPIs**: Speed of deployment from the initial application (application components') selection from the MATILDA Application Repository to the completion of the Application initial deployment on a selected infrastructure. The speed of deployment will be assessed against the relevant 5G-PPP KPI.		
Components	General Solution.		
Testbed	All.		

Detailed Tests Description: Speed of Application Deployment	
Test 1	**Description:** Evaluation of speed of on-boarding process per component and for the whole graph, etc. **Success Criteria:** The speed of the on-boarding process highly depends on the vertical application, with respect to the number of components and their complexity in the resources' definition and handling. The vertical application on-boarding process through the MATILDA interface should take less time than that needed to deploy manually the application graph by using the common open source cloud Infrastructure APIs (i.e. directly through OpenStack APIs).
Result/ Comments	Results have shown that the on-boarding process of vertical applications is considered significantly faster than using the common open source cloud Infrastructure API of OpenStack, since for the non-cloud experts these would require to build expertise in defining interfaces between components, defining the resource requirements, while not having a common view (GUI) of the deployed application graph. As quantified from the results retrieved from the deployment of all MATILDA UC applications, the average components' on-boarding time is about 5-15 min for components and 15-30min for graphs depending on the size of the components and the speed of the remote connection between the verticals local sites and the central repository (as retrieved from [MATILDA-D6.8]-[MATILDA-D6.12]).

Test 2	**Description:** Evaluation of speed of application component/application graph, etc., deployment. **Success Criteria:** The speed of the application component/application graph, etc., deployment should be only restricted by the time needed to deploy application component images on cloud infrastructure (and also by the time needed from the WIM domain to provision the network resources).
Result/ Comments	In principle, the speed of application component/application graph, etc., deployment depends on multiple factors including the time to resolve the slice intent, PoPs resources availability, the specific cloud infrastructure Virtual Machines' spawning time, and so on. As quantified for the applications of the MATILDA UCs, the average deployment time for a single component is ~3 min (capped to the cloud infrastructure Virtual Machines' spawning time) while for an application graph this time is about 15min – 30min, while for first time installation and on-boarding the total time is 90min in line with the global 5G KPIs (as retrieved from [MATILDA-D6.8]-[MATILDA-D6.12]).

5.3 Applications' Lifecycle and Overall Performance Evaluation Aspects

With the completion of the applications' lifecycle management experimentation and testing by the vertical end-users, significant hands-on experience has been obtained. At this stage, key evaluation aspects have been the satisfaction of the main functional and operational requirements of the verticals and the ease of use of MATILDA for the performance of lifecycle management operations. Final performance evaluation results obtained from the verticals'/end-users' perspective corresponding to Test Objectives 1 to 4 of [MATILDA D6.1] have been provided in [MATILDA-D6.8]-[MATILDA-D6.12], and are summarised in this section tables.

Table 15: Bandwidth Allocation Testing.

Number	1	**Type**	End User Performance
Title	Bandwidth Allocation		
Relevant UCs	All.		
Validation method – Tests	Tests to be performed measuring the end-user experienced data rates against the provisioned ones for a number of data sessions (specific tools might be needed) and for various scenarios. Tests will focus on measuring data rates: • Of various provisioned slices (network level), • For the UCs applications (application level), • Under various scenarios requiring bandwidth allocation to be adjusted according to the predefined rules (being specified at application or network level). For bandwidth-intensive scenarios, towards verifying that it is possible to achieve the 5G-PPP bandwidth KPIs once supported by the radio access network.		

KPIs	**KPI:** Data Rates measured in Mbps. **Success Criteria:** It shall be verified that: • The achieved maximum and guaranteed data rates are in-line with the provisioned QoS class of the sub-slices that correspond to specific application component interfaces.
Components	End-to-End.
Testbed	All.

Tests Description: Bandwidth Allocation Testing	
Test 1	**Description:** Evaluation of datarates provisioned for the various links between the application components of an application graph, etc. **Success Criteria:** The achieved maximum and guaranteed data rates are in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces.
Result/ Comments	Results have shown that the achieved maximum and guaranteed data rates are in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces. Depending on the application interface requirements reflected on the slice intent the achieved, guaranteed data rates can range between 10-100Mbps for the MATILDA vertical applications. ([MATILDA-D6.8]- [MATILDA-D6.12]).

Table 16: Network Delay/Latency Testing.

Number	2	**Type**	End User Performance
Title	Network Delay/Latency Testing		
Relevant UCs	All esp. UC1, UC2, UC3.		
Validation method – Tests	Tests to be performed measuring the delay/latency during a number of data sessions (specific tools might be needed). Tests will focus on measuring end-to-end latency between components UCs applications.		
KPIs	**KPI:** Latency measured in ms. **Success Criteria:** It shall be verified that: • The achieved latency is in-line with the required latency restrictions defined in the slice intent.		
Components	End-to-End.		
Testbed	All.		

Tests Description: Network Delay/Latency Testing	
Test 1	**Description:** Evaluation of end-to-end latency between components UCs applications during a number of data sessions. **Success Criteria:** The achieved latencies are in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces.
Result/ Comments	Results have shown that the achieved latencies are in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces. Depending on the application specific requirements reflected on the slice intent the achieved, guaranteed latencies can range between tens to 150ms for the MATILDA vertical applications and for the network technologies materialising the slices in the context of the testbeds implementations. ([MATILDA-D6.8]-[MATILDA-D6.12]).

Table 17: High End-User Device Density Testing.

Number	4	**Type**	Functional
Title	High End-User Device Density		
Relevant UCs	UC5.		
Validation method – Tests	Tests to be performed will focus on ensuring the support of several public lighting poles, introducing a high volume of (IoT signalling) traffic to the network.		
KPIs	**Success Criteria:** 99.99% of messages successfully received.		
Components	End-to-End.		
Testbed	All.		

Tests Description: High End-User Device Density Testing	
Test 1	**Description:** Evaluation of the support of several public lighting poles, introducing a high volume of (IoT signalling) traffic to the network. **Success Criteria:** Support of high volumes of end user devices with high availability 99.99% of messages successfully received and low packet loss.
Result/ Comments	Results have shown that the MATILDA project implementation of the MATILDA framework allows the support of at least 1000 IoT devices (physical or emulated), and that their simultaneous operation can be supported with 0.1% packet loss. As observed 99.99% availability during operation can be also achieved. The results have been presented in more detail in [MATILDA-D6.10].

Table 18: Interoperability.

Number	18	**Type**	Non-Functional
Title	Interoperability with various Access Networks (WAN, LTE, 5G, LoRaWAN/LTE-M, etc.)		
Relevant UCs	UC1, UC2.		
Validation method – Tests	Tests to be performed verifying the MATILDA framework operation over various Access Networks (WAN, LTE, 5G, etc.). In particular, the following will be tested: • Definition of Access technology at MATILDA metamodels and delivery of an end-to-end network slice to the relevant access network nodes; • Evaluation of the QoS definition for the various access network technologies and verification of delivery of a network slice/service with the defined performance/QoS.		
KPIs	**Success Criteria:** Verification of MATILDA framework operation over various Access Networks (WLAN, LTE, 5G, etc.)		
Components	Access network nodes, WIM, VIM, OSS/BSS.		
Testbed	CNIT, ORO, ExxpertSystems premises.		

Tests Description: Interoperability.	
Test 1	**Description:** Evaluation of MATILDA framework operation over various Access Networks (WLAN, LTE, 5G, etc.) **Success Criteria:** Verification of interoperability with various Access Networks (WLAN, LTE, 5G, etc.)
Result/ Comments	Results have shown that the MATILDA project implementation of the MATILDA framework allows the support of multiple access network technologies namely, WLAN and LTE in the case of the Testing 4.0 UC (CNIT - ExxpertSystems premises) demo, and LTE and LTE-M in the case of the smart lighting (ORO) demo. More details can be found in [MATILDA-D6.10] and [MATILDA-D6.12].

6 Adoption Guidelines

Throughout the project lifecycle, over a continuous interaction between the MATILDA core-development team and the vertical use case representatives -providing feedback at various development, integration, testing and evaluation phases-, stakeholders' questions/ remarks/ clarifications/ suggestions / etc. have been collected. All this information has constituted the primary material that evolved to a complete "user guidelines" component, which has been added in VAO in order to provide instantaneous help to the potential end-user.

At the same time, concrete and structured information about the MATILDA framework and a set of installation guidelines for the final release of the MATILDA framework has also been provided in [MATILDA-D5.3], aiming to facilitate the various adopters (e.g. MATILDA demonstrators, interested parties from other 5G PPP projects).

7 Summary - Conclusions

The MATILDA evaluation framework has been specified as flows of validation and evaluation processes spanning from **MATILDA Solution Components and Functionality Validation** to **General (as a whole) Solution Validation and Evaluation** and further to **Performance Evaluation on the basis of specific KPIs** of MATILDA specific functions and of the whole solution. Validation testing and evaluation flows have been addressed at various project stages; namely: at MATILDA component development phases, at MATILDA components' integration phases, at vertical application on-boarding phases, at MATILDA solution operational phases, at vertical application full deployment phases, and finally at vertical application operational phases. Following an iterative development and testing processes scheme, it has been ensured that stakeholders' clarifications/suggestions/changes are addressed. Along these lines, as planned, the list of test objectives and procedures have been refined throughout the project lifetime to better suit implementation specificities that emerge in these project stages, along with testbed specific features, environment setup/tools, etc.

To this end, testing and validation activities have been performed for the following MATILDA components/functionalities:

1. The Application Development and Wrapping Toolkit,
2. The MATILDA Marketplace; in particular, the Application and VNF repositories,
3. The VAO,
4. The new Telecom Layer, including the OSS, with special focus on 3GPP network services management,
5. The NFVO,
6. The VIM,
7. The WIM, and
8. The Wide-Area SDN Controller (WSC).

With the complete integration of these components, workflow-based testing has been performed, namely:

- testing of the application orchestration functionality ranging from application components & service graphs definition to its deployment on various VIMs through VAO,
- testing of application deployment re-configuration, namely scaling capabilities triggered by various factors,
- testing of the application deployment functionality along with network functions deployment through VAO and NFVO, complemented with 3GPP network service provisioning and lifecycle management of telecom services through the MATILDA new telecom layer (OSS/BSS),

- testing of multisite resource management functionality, including various VIMs and WIM,
- testing of the Wide-Area SDN network control functionalities.

The results retrieved with the completion of the validation and evaluation activities, are summarised as follows:

- The Application development and wrapping functionalities (to make an application 5G-ready) have been completely developed and tested (at UBITECH/CNIT testbeds, using all MATILDA Demonstrators' Applications), in terms of successfully enabling:
  - creation/edition/modification of application service graphs adhering to the MATILDA metamodels,
  - creation/edition of runtime policies (in particular resource utilization and security – related ones) at application component level.
- The lifecycle management (insertion, modification/update, selection, deletion) of applications/application components/VNFs and their metadata in the associated repositories has been tested and successfully validated.
- The Vertical Applications' orchestration and lifecycle management has been tested (at UBITECH/CNIT/Demonstrator testbeds), in terms of successfully enabling:
  - extraction of the slice intent from the service graphs definitions on the basis of the MATILDA metamodels,
  - Real-Time deployment of an application in various VIMs/PoPs,
  - enforcement of specific run-time policies (resource utilisation, and security-related),
  - termination of application instance operation upon request.
- A number of the Vertical Applications deployment monitoring functionalities have been tested (at UBITECH/CNIT/Demonstrator testbeds), and the following have been successfully validated:
  - Activation/configuration of active and passive probes monitoring compute/ network/ application resources utilisation/behaviour,
  - Fusion of monitoring data coming from multiple parallel data loads from multiple sources,
  - Extraction of Real-Time Analytics and advanced insights.
- The lifecycle management of NSs has been completely developed and tested (at CNIT/ ORAN/ ININ testbeds), in terms of successfully enabling:
  - Lifecycle management of a 3GPP network services slice,
  - MEC capabilities using a "Bypass VNF" enabling traffic offloading at edge VIMs/PoPs,
  - WSC – specific functionality related to the management of network resources on a per slice basis.

Following, the MATILDA component and functionality and workflow-based validation and evaluation, complete demonstrations have been made with all MATILDA vertical applications, and significant hands-on experience has been acquired by the verticals as MATILDA end-users. As revealed from the experimentation, testing and evaluation results that have been collected from all demonstrators:

- The on-boarding, application deployment and lifecycle management processes of vertical applications have been considered as user-friendly and adequate in terms of network service definition and procedural steps for the partners/personnel involved in the project.
- The average deployment time has been significantly minimised, especially when considering the second / third /etc. time deployment, being for a single component is ~3 min (capped to the cloud infrastructure Virtual Machines' spawning time) while for an application graph this time is about 15min – 30min, while for first time installation and on-boarding the total time is 90min inline with the global 5G KPIs.
- QoS guarantees can be provisioned in terms of achieved maximum and guaranteed data rates, in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces. Depending on the application interface requirements reflected on the slice intent the achieved, guaranteed data rates can range between 10-100Mbps for the MATILDA vertical applications.
- QoS guarantees can be provisioned in terms of achieved latencies, in-line with those defined in the slice intent and negotiated with the OSS/BSS and finally provisioned to the sub-slices between the relevant application component interfaces. Depending on the application specific requirements reflected on the slice intent the achieved, guaranteed latencies can range between tens to 150ms for the MATILDA vertical applications and for the network technologies materialising the slices in the context of the testbeds implementations.
- High number of connections can be supported over the MATILDA project implementation of the MATILDA framework allowing the support of at least 1000 IoT devices (physical or emulated), and that their simultaneous operation can be supported with 0.1% packet loss, and 99.99% availability during operation.

Last but not least, through a continuous interaction between the MATILDA core-development team and the vertical use case representatives -providing feedback at various development, integration, testing and evaluation phases-, stakeholders' questions/ remarks/ clarifications/ suggestions / etc. have been collected. The latter have constituted the primary material that evolved to a complete "user guidelines" component, which has been added in VAO in order to provide instantaneous help to the potential end-user.

References

[5GPP-2015] 5GPPP, "5G Vision: The next generation of communication networks and services". URL: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf

[MATILDA-D1.1] MATILDA project deliverable, D1.1 - MATILDA Framework and Reference Architecture, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D1.5] MATILDA project deliverable, D1.5 - Deployment and Runtime Policy Metamodel, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D4.1] MATILDA project deliverable, D4.1 – Multi-site Resources Management and Execution Mechanisms.

[MATILDA-D4.2] MATILDA project deliverable, D4.2 – Network and Computing Slice, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.1] MATILDA project deliverable, D6.1 - Evaluation Framework and Demonstrators' Planning, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.2] MATILDA project deliverable, D6.2 - Emergency Infrastructure with SLA Enforcement Implementation Report, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.3] MATILDA project deliverable, D6.3 - High Resolution Media on Demand Implementation Report, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.4] MATILDA project deliverable, D6.4 - Smart City Intelligent Lighting System Implementation Report, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.5] MATILDA project deliverable, D6.5 - Industry 4.0 Smart Factory Implementation Report, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.6] MATILDA project deliverable, D6.6 - Automobile Electrical Systems Remote Control Implementation Report, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.7] MATILDA project deliverable, D6.7 – Validation Results, Performance Evaluation and Adoption Guidelines, available online at: http://www.MATILDA-5g.eu/index.php/outcomes

[MATILDA-D6.8] MATILDA project deliverable, D6.8 - Emergency Infrastructure with SLA Enforcement Implementation Report

[MATILDA-D6.9] MATILDA project deliverable, D6.9- High Resolution Media on Demand Implementation Report

[MATILDA-D6.10] MATILDA project deliverable, D6.10 - Smart City Intelligent Lighting System Implementation Report

[MATILDA-D6.11] MATILDA project deliverable, D6.11 - Industry 4.0 Smart Factory Implementation Report

[MATILDA-D6.12] MATILDA project deliverable, D6.12 - Automobile Electrical Systems Remote Control Implementation Report

[TS 22.261] 3GPP TS 22.261 V16.1.0 (2017-09) 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Service requirements for the 5G system; Stage 1 (R16)

[TR 22.861] 3GPP, TR 22.861, "Feasibility Study on New Services and Markets Technology Enablers for massive Internet of Things; Stage 1"

[TR 22.862] 3GPP, TR 22.862, "Feasibility study on new services and markets technology enablers for critical communications; Stage 1"

[TR 22.863] 3GPP, TR 22.863, "Feasibility study on new services and markets technology enablers for enhanced mobile broad-band; Stage 1"

[TR 22.864] 3GPP, TR 22.864, "Feasibility study on new services and markets technology enablers for network operation; Stage 1"

[TR 22.891] 3GPP, TR 22.891, "Feasibility study on new services and markets technology enablers; Stage 1"

[ITU-R M.2083] Recommendation ITU-R M.2083-0 (09/2015): IMT Vision - Framework and overall objectives of the future development of IMT for 2020 and beyond

[NGMN 5G] NGMN 5G Initiative Team, "A Deliverable by the NGMN Alliance", NGMN 5G White Paper, Feb.17, 2015 - https://www.ngmn.org/5g-white-paper/5g-white-paper.html

[5G PPP PMR] 5G PPP, 5G PPP phase II KPIs – Annex to Programme Management Report, August 2019.

[5GPPP Broch] 5G empowering vertical industries, brochure available online at https://5g-ppp.eu/wp-content/uploads/2016/02/BROCHURE_5PPP_BAT2_PL.pdf

[DU-BRUSCHI] "K.X. Du, B. Sayadi, G. Carrozzo, F. Lazarakis, A. Kourtis, M.S. Siddiqui, J. Sterle, O. Carrasco, and R. Bruschi, "Definition and Evaluation of Latency in 5G: A Framework Approach", URL: http://www.jkjmanagement.com/5gwf19-4/papers/p135-du.pdf, 2019 IEEE 5G World Forum"